Generating Defeasible Knowledge Bases from Real-World Argumentations using D-BAS[⋆]

Daniel Neugebauer

Computer Science Institute, Heinrich-Heine University Düsseldorf,
neugebauer@cs.uni-duesseldorf.de

Abstract. D-BAS is an open-source web tool for dialog-based online argumentation among non-expert human users [7]. In this work, we present DABASCO, a D-BAS module that allows to automatically export D-BAS discussions, interpreted as defeasible knowledge bases, into formats of three well-established argumentation models: abstract Argumentation Frameworks, the ASPIC Framework, and Abstract Dialectical Frameworks.

1 Introduction

A major challenge of real world application of formal models of arguments is to establish a "pipeline" that allows to seamlessly map real argumentation data to formal model instances. A large body of research concentrates on developing methods to annotate natural language argumentations in order to identify their logical structure [13]. D-BAS [7], as opposed to typical tools for online argumentation like forums and comment sections, has the upside of not generating unstructured bodies of natural language texts. Users can input text only in the form of short *statements*, which hold no internal logical structure. Logical relations between statements are given on a purely meta level using the Web interface. Therefore, D-BAS discussions are automatically, naturally structured.

The purpose of this work is to present DABASCO, a tool that utilizes D-BAS as a source of structured argumentation data and allows to translate D-BAS data into abstract Argumentation Frameworks [4], the ASPIC framework [3], and Abstract Dialectical Frameworks [2]. We briefly describe the D-BAS data model, then give short definitions of the target argumentation models that DABASCO translates to, demonstrate the translations, and provide instructions on how to download and use DABASCO.

2 D-BAS Discussions as Defeasible Knowledge Bases

A D-BAS discussion consists of a set of atomic *statements*, a set of *arguments*, and a user *opinion* for each user of the system. Each argument consists of a set of statements as the argument's *premise* and a single (possibly negated) statement

⋆ This work was supported in part by the project "Online Participation," funded by the NRW Ministry for Innovation, Science, and Research.

or a single negated argument as its *conclusion*. The argument represents the application of an *inference rule* which claims that the premise is a reason to believe the conclusion. Each user opinion holds information about which statements and arguments that user has accepted or rejected. For more detailed information, we refer to Krauthoff et al. [7].

Example 1 (The Nixon Diamond as a D-BAS *discussion).* To illustrate the syntax of a D-BAS discussion, we formalize the well-known Nixon diamond as a D-BAS discussion. Each statement and inference rule is given with its corresponding ID, where rule IDs have a leading "r" to avoid confusion.

Statements:

1 "Nixon is a pacifist"
2 "Nixon is a quaker"
3 "Nixon is a republican"

Inference Rules:

$r1$ "Nixon is a pacifist" *because* "Nixon is a quaker".
$r2$ "Nixon is **not** a pacifist" *because* "Nixon is a republican".

In Example 1, the two arguments that apply inference rules $r1$ and $r2$, respectively, *rebut* each other—they have conflicting conclusions. Following Prakken [8], D-BAS allows two more variants of attack among arguments: *undermining* attacks, where an argument's conclusion contradicts a premise statement of another argument, and *undercutting* attacks, where an argument has the negated identifier of another argument as its conclusion, i.e., directly attacks the inference rule applied in that other argument. To showcase these types of argument attack, we slightly extend Example 1 by the following (fictitious) statements and arguments.

Example 1 (continued). We add inference rules $r3$ and $r4$ with two new statements 4 and 5 to the discussion:

$r3$ "Nixon is **not** a quaker ($\neg 2$)" *because* "Nixon converted to Catholicism (4)".
$r4$ *Rule* $r2$ *does not apply because* "there are pacifist republicans (5)".

Disregarding the content of the statements yields the following abstract representation of the discussion, which will be used as a running example throughout the paper:

$$statements = \{1, 2, 3, 4, 5\}$$
$$inferences = \{r1 : [2] \Rightarrow 1, r2 : [3] \Rightarrow \neg 1, r3 : [4] \Rightarrow \neg 2, r4 : [5] \Rightarrow \neg r2\}$$

A D-BAS discussion can be seen as a *defeasible knowledge base* (DKB) [3]. A DKB consists of a set of *literals* (closed under negation) and a set of *inference rules* that indicate logical relations between literals. Inference rules have a *body* and a *head*, where the body is a conjunction of literals and the head is a single literal, indicating that the body infers the head. A rule body may be empty, in which case there is no precondition to infer the head. The inference rules are often divided into *strict* and *defeasible* rules, where the head of a strict rule is always true if the body is true, whereas there may be circumstances where the head of a defeasible rule is not true even when its body is.

A D-BAS discussion can be mapped to a DKB by identifying D-BAS statements and their negations with literals and D-BAS inference rules as defeasible inference rules. Since all arguments in D-BAS are user-generated, there are no strict rules. To cover undercutting attacks, it is also required to allow rule identifiers as heads of defeasible rules, either by including rule identifiers in the set of literals or by directly allowing meta-level inferences. Further, user opinions can be used as a source for *assumptions*. Some argumentation systems explicitly model assumptions (e.g., the ASPIC framework [3]). In other systems, they can be implemented in the form of strict or defeasible rules with an empty body and the literal itself as the head.

3 Translations

D-BAS in itself is not restricted to a specific normative interpretation, since all data is created by subjective users. This allows to interpret D-BAS data in many different ways. DABASCO currently implements translations of D-BAS discussions to Argumentation Framework (AF), ASPIC, and Abstract Dialectical Framework (ADF) representations. Both the ASPIC and the ADF translation evaluate a single user opinion in the context of the dicussion by using it as a source for assumptions. The AF translation does not employ a user opinion and therefore produces an "objective" representation of the discussion.

3.1 Abstract Argumentation Frameworks

An Argumentation Framework (AF) is a pair $\langle \mathcal{A}, \mathcal{R} \rangle$ where $\mathcal{A}$ is a finite set of *arguments* and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is a binary *attack relation* on the arguments. Wyner et al. [12] proposed a method to translate defeasible knowledge bases to abstract AFs, which was implemented by Strass [11] for standard DKBs. DABASCO provides an implementation that, in addition, covers the undercutting inference rules which are possible in D-BAS discussions, without encoding inference rule identifiers as additional literals: in the translation, each argument representing an undercutting rule simply attacks its target argument. Other than that, the translation implemented in DABASCO is exactly that of Wyner et al.—for a description, please refer to the original paper. DABASCO produces AFs in ASPARTIX syntax [5] that can be directly fed into most existing AF solvers.

Example 2. Figure 1 displays the argumentation framework obtained by translating the discussion from Example 1. The generated AF has 34 preferred extensions. All arguments are credulously acceptable and none are skeptically acceptable for the preferred semantics, indicating that all statements can be defended and all rules can be activated.

3.2 Abstract Dialectical Frameworks

An Abstract Dialectical Framework (ADF) is a triple $D = (S, L, C)$, where S is a set of *statements*, $L \subseteq S \times S$ is a set of directed *links* between statements,

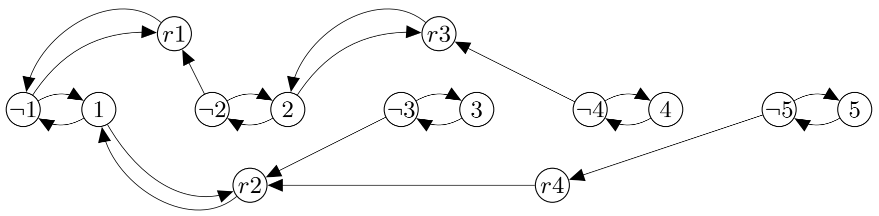

Fig. 1. AF created from Example 1. Arguments are represented as nodes, arrows symbolize the attack relation.

and $C = \{C_s\}_{s \in S}$ is a set of *acceptance conditions* for each statement. Each acceptance condition C_s can be represented as a boolean formula over the acceptance status of all parents (with respect to L) of s. DABASCO implements a translation of a D-BAS discussion including a single user opinion to an ADF following a translation proposed by Strass [10]—again, we refer to the original paper for details. DABASCO uses an extended DKB for the translation: for each accepted or rejected statement in the given user opinion, a bodyless strict rule is added which asserts that acceptance/rejection. These strict rules enforce the user opinion in the ADF. Without such a starting point for reasoning, no acceptance for any statement can be derived and all inference rules remain trivially inactive, because their premises do not hold. Since D-BAS discussions hold no facts, we figured user opinions to be the most reasonable source for such "grounded" knowledge. DABASCO produces ADFs in a clause-based format suitable to be used as input for the DIAMOND [6] and YADF [1] solvers.

Example 3. Consider a user opinion in which statements 3 and 4 are accepted and statement 5 is rejected. The following acceptance functions are generated when applying the ADF translation to the discussion from Example 1 using this user opinion, where s_i, i_j and a_k encode statement i, inference rule j and user assumption k, respectively, and a leading n indicates negation:

$$\begin{aligned}
C_{s_1} &= \neg C_{ns_1} \land C_{i_1} & C_{i_1} &= \neg C_{ns_1} \land \neg C_{ni_1} \land C_{s_2} & C_{a_3} &= \texttt{true} \\
C_{ns_1} &= \neg C_{s_1} \land C_{i_2} & C_{ni_1} &= \neg C_{i_1} & C_{na_3} &= \neg C_{s_3} \land \neg C_{na_3} \\
C_{s_2} &= \texttt{false} & C_{i_2} &= \neg C_{s_1} \land \neg C_{ni_2} \land C_{s_3} & C_{a_4} &= \texttt{true} \\
C_{ns_2} &= \neg C_{s_2} \land C_{i_3} & C_{ni_2} &= \neg C_{i_2} & C_{na_4} &= \neg C_{s_4} \land \neg C_{na_4} \\
C_{s_3} &= \neg C_{ns_3} \land C_{a_3} & C_{i_3} &= \neg C_{s_2} \land \neg C_{ni_3} \land C_{s_4} & C_{a_5} &= \texttt{true} \\
C_{ns_3} &= \texttt{false} & C_{ni_3} &= \neg C_{i_3} & C_{na_5} &= \neg C_{ns_5} \land \neg C_{na_5} \\
C_{s_4} &= \neg C_{ns_4} \land C_{a_4} & C_{i_4} &= \neg C_{i_2} \land \neg C_{ni_4} \land C_{s_5} & & \\
C_{ns_4} &= \texttt{false} & C_{ni_4} &= \neg C_{i_4} & & \\
C_{s_5} &= \texttt{false} & & & & \\
C_{ns_5} &= \neg C_{s_5} \land C_{a_5} & & & &
\end{aligned}$$

The ADF has the following four preferred models (*undecided* indicates cases where both an ADF statement and its negation are rejected):

	model 1	model 2	model 3	model 4
C_{s_1}	*undecided*	`false`	*undecided*	`false`
C_{s_2}	`false`	`false`	*undecided*	*undecided*
C_{s_3}	`true`	`true`	`true`	`true`
C_{s_4}	`true`	`true`	`true`	`true`
C_{s_5}	`false`	`false`	`false`	`false`
C_{i_1}	`false`	`false`	`false`	`false`
C_{i_2}	`false`	`true`	`false`	`true`
C_{i_3}	`true`	`true`	`false`	`false`
C_{i_4}	`false`	`false`	`false`	`false`
C_{a_3}	`true`	`true`	`true`	`true`
C_{a_4}	`true`	`true`	`true`	`true`
C_{a_5}	`true`	`true`	`true`	`true`

Table 1. Preferred models in the ADF representation of Example 1.

Statements 1 and 2 are both rejected in two models and undecided in two models. Statements 3 and 4 are accepted in all models and statement 5 is rejected in all models (in compliance with the user opinion). Rules $r1$ and $r4$ are inactive in all models, and rules $r2$ and $r3$ are both active in two models.

3.3 The ASPIC Framework

The ASPIC framework is a very powerful formalism that allows to directly encode DKBs. It is expressive enough to represent all features of a D-BAS discussion without the need for a translation. DABASCO generates ASPIC instantiations that are formatted as input for the TOAST online ASPIC solver [9].

Example 4. We export the extended Nixon example with the same user opinion as in the previous example. TOAST creates five arguments:

$$"A1 : 3", \quad "A2 : 4", \quad "A3 :\sim 5", \quad "A4 : A2 \Rightarrow\sim 2", \quad "A5 : A1 \Rightarrow\sim 1"$$

It produces a single preferred extension in which statements 3 and 4 are acceptable and statements 1, 2 and 5 are not.

4 Using dabasco

DABASCO is available for download at `https://github.com/hhucn/dabasco`. It requires python 3 to run. A full guide for setup and use is included in the repository, along with a small mockup app that serves example D-BAS data and allows to test DABASCO's features without running D-BAS. All Examples in this paper can be verified using discussion 2 and user 1 from the mockup app. DABASCO is published under the MIT license and can be freely used and distributed.

References

1. Brewka, G., Diller, M., Heissenberger, G., Linsbichler, T., Woltran, S.: Solving advanced argumentation problems with answer-set programming. In: Proceedings of the 31th AAAI Conference on Artificial Intelligence. pp. 1077–1083 (2017)
2. Brewka, G., Woltran, S.: Abstract dialectical frameworks. In: Principles of Knowledge Representation and Reasoning: Proceedings of the 12th International Conference. AAAI Press (2010)
3. Caminada, M., Amgoud, L.: On the evaluation of argumentation formalisms. Artificial Intelligence 171(5), 286–310 (2007)
4. Dung, P.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321–357 (1995)
5. Egly, U., Gaggl, S., Woltran, S.: Answer-set programming encodings for argumentation frameworks. Argument & Computation 1(2), 147–177 (2010)
6. Ellmauthaler, S., Strass, H.: The diamond system for computing with abstract dialectical frameworks. In: Proceedings of the 5th International Conference on Computational Models of Argument. pp. 233–240 (2014)
7. Krauthoff, T., Baurmann, M., Betz, G., Mauve, M.: Dialog-based online argumentation. In: Proceedings of the 6th International Conference on Computational Models of Argument. pp. 33–40. IOS Press (2016)
8. Prakken, H.: An abstract framework for argumentation with structured arguments. Argument & Computation 1(2), 93–124 (2010)
9. Snaith, M., Reed, C.: Toast: Online aspic+ implementation. Proceedings of the 4tg International Conference on Computational Models of Argument 245, 509–510 (2012)
10. Strass, H.: Instantiating knowledge bases in abstract dialectical frameworks. In: Computational Logic in Multi-Agent Systems, pp. 86–101. Springer (2013)
11. Strass, H.: Implementing instantiation of knowledge bases in argumentation frameworks. In: Proceedings of the 5th International Conference on Computational Models of Argument. pp. 475–476 (2014)
12. Wyner, A., Bench-Capon, T., Dunne, P.: On the instantiation of knowledge bases in abstract argumentation frameworks. In: International Workshop on Computational Logic in Multi-Agent Systems. pp. 34–50. Springer (2013)
13. Wyner, A., van Engers, T., Hunter, A.: Working on the argument pipeline: Through flow issues between natural language argument, instantiated arguments, and argumentation frameworks. Argument & Computation 7(1), 69–89 (2016)