Legal Knowledge and Information Systems
G. Sileno et al. (Eds.)

doi:10.3233/FAIA230994

Answer Set Programming for Legal Decision Support and Explanation

Daniele THESEIDER DUPRÉ [1]
DISIT, Università del Piemonte Orientale
ORCiD ID: Daniele Theseider Dupré https://orcid.org/0000-0001-6798-4380

Abstract. The ANGELIC methodology was successfully used to predict decisions of the European Court of Human Rights based on a set of logical rules, with significantly better accuracy than the one achieved by machine learning approaches, as well as to explain the results of reasoning, quite valuable in order to make them trustworthy. This work demonstrates a different logic-based approach, based on Answer Set Programming for solving and generating explanations for solutions. The use of a general knowledge representation and reasoning system, where representation and inference are not tightly coupled, allows for using the same representation for inference tasks different from prediction, thus getting more value out of the domain model, and opens for integrating further forms of knowledge.

Keywords. Answer Set Programming, Reasoning with Legal Cases, Computational Models of Argument, Explanation.

Introduction

The ANGELIC methodology was defined [1,2] for deciding legal cases, based on computational models of arguments, and, in particular, Abstract Dialectical Frameworks (ADFs) [3] to represent the *issues* and *factors* to be taken into account and their relations (such ADFs are part of the ANGELIC Domain Models in [2]).

The approach has been applied [4,5] for reproducing the decisions of the European Court of Human Rights (ECtHR) related to a specific article (Art.6) of the European Convention on Human Rights (ECHR), about the right to a fair trial. Some effort was required in developing the representation, but the resulting system achieved a high accuracy (97%) in prediction, compared with a 70-85% of machine learning approaches.

An even more important advantage lies in the fact that the structured representation in the ADF, expressed in terms of intermediate concepts that are important for humans in relating inputs to a conclusion, is used as a basis for *explanation* of the results of reasoning, which is of utmost importance for making an automated system acceptable and trustworthy for users in this domain, as well as in other ones, and would be difficult to achieve using machine learning.

In [4] Prolog programs are used; inference and explanation are achieved relying on the backtracking execution model. Then the program is not exactly a model of the domain, while it works well for the purpose of estabilishing the conclusion in a case

[1] Corresponding Author: Daniele Theseider Dupré, dtd@uniupo.it.

and for providing a trace of reasoning in natural language. A different implementation is provided in [5], where a Java program (rather than a general reasoning system), used for interpreting the specific form of ADF considered, provides the back end for a web-based interface for explanation.

In this paper (an abridged version of [6]) we propose a different approach where:

- Answer Set Programming (ASP) [7] is used for modelling the domain, so that different forms of reasoning can be provided on the same model as Answer Set Solving: reasoning that is not just predicting the outcome from a complete set of inputs (*base level factors*), but may include, e.g., how to complete the input, or to modify the input, in order to achieve a given outcome;
- information for explanation is associated with the representation, but not part of it; explanation may be provided by suitable extensions of solvers and/or visualization tools (e.g., *xASP2* [8], *clingraph* [9]); in this paper we consider the *xclingo* [10] extension of the *clingo* ASP solver, where annotations of rules and facts can be used to provide a tree-like explanation.

1. Modeling, Reasoning and Explaining in the Legal Domain with ASP

In the following we consider the form of ADF used in [1,4], where a tree is provided for a main decision (tree root), other nodes are issues and factors relevant to the decision, the leaves being *base level factors* whose truth has to be established in ways that are not provided in this version of the model. For internal nodes, a *sequence* of rules of the form: `Accept/Reject Node if Condition`, is given, with conditions in terms of children of the node; a rule can be applied if the previous ones cannot; the last rule has an empty precondition, then providing a default boolean value for `Node`.

We consider the fragment of ADF in [4], table 1 (with some correction based on [5]), with the tree structure in figure 1. The factors for which no rule is provided in this fragment (tree leaves) are considered here base level ones. An ASP model for the ADF can be provided, using explicit negation (to correspond to the valid/invalid factors in [4]), default negation, and, where useful, additional atoms to represent that a previous rule does not apply.

The model can be used for several reasoning tasks, combining it with a complete or incomplete *case description* - intended as an assignment of boolean values to base level factors; in order to consider both alternative values for some (possibly all) of them, *choice rules* of ASP are used. The reasoning tasks include:

- *Prediction* of the decision from a complete case description.
- *Finding case descriptions* that satisfy some given condition, e.g., a case that is predicted to be admissible, but for which no violation of Art. 6 is predicted. This may be of interest for analyzing the domain model, also for education purposes.
- *Reasoning on incomplete case descriptions*, to find whether a decision can be predicted from the known values of base level factors; or to find base level factors to provide support for, in order to achieve a desired prediction (esp. a "violation" prediction). This may be of interest for lawyers being consulted on the possible submission to the Court of cases where not all evidence is already available.

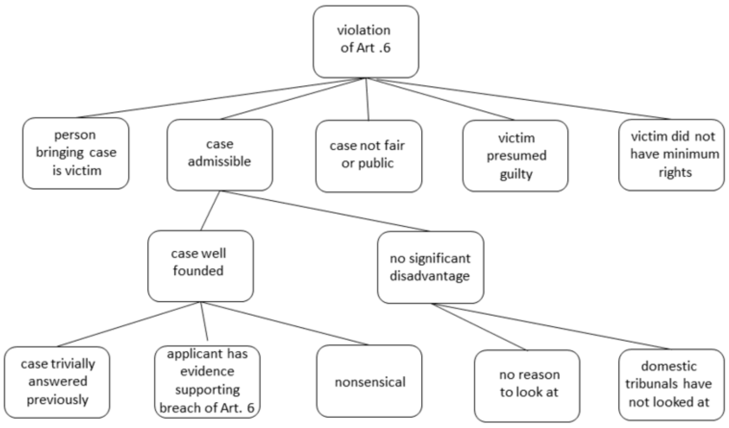
violation of Art .6
person bringing case is victim
case admissible
case not fair or public
victim presumed guilty
victim did not have minimum rights
case well founded
no significant disadvantage
case trivially answered previously
applicant has evidence supporting breach of Art. 6
nonsensical
no reason to look at
domestic tribunals have not looked at

Figure 1. Tree structure of the ADF fragment

- Finding a minimum number of changes in the truth of base level factors in order to change the prediction in a case, e.g. from “no violation” to “violation”, a form of *counterfactual reasoning* which would provide an additional form of explanation, with respect to tracing a prediction result (considered below).

1.1. Explanation

The *xclingo* system[2] [10] provides ways for producing explanations for the solutions of an ASP solver, in the form of trees built from annotations of some of the atoms/rules in the ASP model. In this paper we rely on atom annotations, which can be kept separate from the model itself (e.g. in a separate file); a `%!show_trace` annotation for an atom means that, if the corresponding atoms is in a solution, we are interested in explaining how it is derived; `%!trace` annotations are used for the atoms corresponding to the nodes in the ADF, to mean that if they are used in a proof, the associated string should be included in the tree-like explanation.

The following are the explanations obtained for the predicted decision in three case descriptions; in each of them a different group of reject/accept clauses (in the last one, a default “reject” rule) is applied for the root node of the ADF.

```
*
|__No violation of art. 6
|  |__the case is not admissible
|  |  |__there was no significant disadvantage
|  |  |  |__there are domestic tribunals that have not looked at the case

*
|__Violation of art. 6
|  |__the case is admissible
|  |__the person bringing the case is the victim
|  |__the case was not fair and public
```

[2]https://github.com/bramucas/xclingo

```
*
|__No violation of art. 6
|  |__the case is admissible
|  |__the person bringing the case is the victim
|  |__the case was fair and public
|  |__the victim was not presumed guilty
|  |__the victim had the minimum rights
```

2. Conclusions

We have sketched how legal reasoning with tree-form textual explanation can be achieved in Answer Set Programming. This is seen as a step in order to achieve results similar to the ones in [4,5] with some advantages. As usual, the availability of a model of a domain allows for reasoning that is not just predicting an output for a complete set of inputs. This would make the effort of developing the domain model even more worthwhile. Other ASP tools for explanation and visualization can be used to provide further ways for presenting the results to non-technical users. Moreover, using a general representation and reasoning framework like ASP could lead to integrating knowledge from ADFs with possible generalizations of them, and other forms of knowledge, without implementing reasoners for the purpose.

References

[1] Al-Abdulkarim L, Atkinson K, Bench-Capon TJM. A methodology for designing systems to reason with legal cases using Abstract Dialectical Frameworks. Artif Intell Law. 2016;24(1):1-49. Available from: https://doi.org/10.1007/s10506-016-9178-1.

[2] Atkinson K, Bench-Capon TJM. ANGELIC II: An Improved Methodology for Representing Legal Domain Knowledge. In: Proceedings ICAIL 2023. ACM; 2023. p. 12-21.

[3] Brewka G, Strass H, Ellmauthaler S, Wallner JP, Woltran S. Abstract Dialectical Frameworks Revisited. In: IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence. IJCAI/AAAI; 2013. p. 803-9.

[4] Collenette J, Atkinson K, Bench-Capon TJM. An Explainable Approach to Deducing Outcomes in European Court of Human Rights Cases Using ADFs. In: Computational Models of Argument, COMMA 2020. vol. 326 of Frontiers in Artificial Intelligence and Applications. IOS Press; 2020. p. 21-32. Available from: https://doi.org/10.3233/FAIA200488.

[5] Collenette J, Atkinson K, Bench-Capon TJM. Explainable AI tools for legal reasoning about cases: A study on the European Court of Human Rights. Artif Intell. 2023;317:103861. Available from: https://doi.org/10.1016/j.artint.2023.103861.

[6] Theseider Dupré D. Explainable Answer Set Programming for Legal Decision Support. In: Proc. of the 38th Italian Conference on Computational Logic. vol. 3428 of CEUR Workshop Proceedings; 2023.

[7] Brewka G, Eiter T, Truszczynski M. Answer set programming at a glance. Commun ACM. 2011;54(12):92-103.

[8] Alviano M, Trieu LL, Son TC, Balduccini M. Advancements in xASP, an XAI System for Answer Set Programming . In: Proc. of the 38th Italian Conference on Computational Logic. vol. 3428 of CEUR Workshop Proceedings; 2023.

[9] Hahn S, Sabuncu O, Schaub T, Stolzmann T. Clingraph: ASP-Based Visualization. In: Logic Programming and Nonmonotonic Reasoning - 16th International Conference, LPNMR 2022. vol. 13416 of Lecture Notes in Computer Science. Springer; 2022. p. 401-14.

[10] Cabalar P, Fandinno J, Muñiz B. A System for Explainable Answer Set Programming. In: ICLP Technical Communications 2020. vol. 325 of EPTCS; 2020. p. 124-36. Available from: https://doi.org/10.4204/EPTCS.325.19.