Explainability through argumentation in logic programming

Marco Billi[1], Roberta Calegari[1], Giuseppe Contissa[1], Giuseppe Pisano[1], Galileo Sartor[2] and Giovanni Sartor[1]

[1]*Alma AI – Alma Mater Research Institute for Human-Centered Artificial Intelligence, Alma Mater Studiorum—Università di Bologna, Italy*
[2]*University of Torino, Torino, Italy*

Abstract

The paper discusses how explainability can be provided through logic programming and argumentation. In particular, a real case study is considered – the CrossJustice project – together with a ready-to-use technology—namely Arg2P, which seamlessly integrates logic programming and argumentation. Examples are developed and discussed showing the effectiveness of such a blended approach.

Keywords

explainability, logic programming, argumentation, Arg2P, legal reasoning

1. Introduction

Declarative, logic-based approaches represents an alternative way of delivering intelligence, complementary to the one pursued by sub-symbolic approaches [1, 2]. In particular, logic-based technologies may address opaqueness issues, and, when suitably integrated with argumentation capabilities, can support features like interpretability, observability, accountability, and explainability [3, 4, 5].

When merging logic-based approaches – such as logic programming – with argumentation, explainability can be achieved by implementing functions for debating about situations and choices, and for providing reports and insights concerning the case or the decision at hand. An explanation can be seen as a sort of conversation among the person asking for clarifications and the system, enabled by the fact that the system is capable of answering and arguing about the questions [4, 6].

The focus of this paper is to highlight how argumentation and logic programming can support explainability, in particular by appropriately injecting logical inference and argumentation processes when and where required (also in already existing systems). Providing explanation

CAUSAL'21: Workshop on Causal Reasoning and Explanation in Logic Programming, September 20–27, 2021, Rende, Italy

marco.billi3@unibo.it (M. Billi); roberta.calegari@unibo.it (R. Calegari); giuseppe.contissa@unibo.it (G. Contissa); g.pisano@unibo.it (G. Pisano); galileo.sartor@unito.it (G. Sartor); giovanni.sartor@unibo.it (G. Sartor)
0000-0002-6807-073X (M. Billi); 0000-0003-3794-2942 (R. Calegari); 0000-0002-8511-1505 (G. Contissa); 0000-0003-0230-8212 (G. Pisano); 0000-0001-6355-851X (G. Sartor); 0000-0003-2210-0398 (G. Sartor)

CEUR Workshop Proceedings (CEUR-WS.org)

functionalities, and grounding them on arguments, enables users to understand the rationale of a decision, assessments or predictions, so as to choose whether trust them, disregard them, or even inhibit the same outcomes in future cases [7].

Accordingly, the paper discusses how argumentation based on a logic programming approach can be used for justifications and explanation. An existing scenario is taken as a testbed – the CrossJustice project – and experiments are drawn with the Arg2P technology. In this context, Arg2P [8, 9] meets eXplainable Artificial Intelligence (XAI) requirements by providing a light-weight logic-programming argumentation engine.

Within the CrossJustice project that deals with the rights of accused individuals in different EU jurisdictions, argumentation helps to answer the following questions: Which elements shall be taken into account in order to identify the applicable rights? Which elements shall be analysed in order to support the reasoning about such rights?

2. Case study: the CrossJustice project

The CrossJustice project studies the rights of defendants in criminal matters according to the laws of several EU Member States and provides a decision support system, accessible by professionals and citizens alike. The system provides assessments concerning specific cases, and moreover determines the level of harmonisation of national legislations, namely the extent to which national legal frameworks and regulatory acts are in line with the EU acquis and relevant legislative acts of the European Union.

The aim is to help legal practitioners in their daily activities, through a platform that supports interoperability and communication between the several legislative measures that single member states have adopted, by showing how these laws interact, and the extent to which they comply with EU law.

The main target of the project is the creation of a rule-based expert system including a computable representation of the European directives[1] related to the rights of suspects and accused persons in criminal proceedings and the national articles concerning the same subject matter. The system shall inform the users of all the applicable rights according to both European and national acts.

All articles in such Directives and the relevant portions of national transpositions have been represented in Prolog. The legal analysis, by expert lawyers, has often involved the interpretation of complex legal rules, and the reconstruction of the dependencies between norms. The main advantage of such a symbolic representation lies in its understandability both to human programmers and to legal experts, and on the possibility to track the reasoning processes.

Legal professionals, after verifying whether a right itself exists, are interested in the conditions under which the right is granted, and on the measures available for its protection. The purpose of the aforementioned directives is to provide a clear guideline for the national legislators to follow and subsequently create concrete plans to execute and provide the rights to their citizens. As single Member States have discretionary power in implementing these provisions, they may

[1] Directive 2016/343, Directive 2010/64, Directive 2016/800, Directive 2016/1919, Directive 2012/13, Directive 2013/48

adapt the text of national provisions accordingly. This may result in discrepancies that may give rise to a wide spectrum of legal issues.

The aim of the system is thus twofold. First, it shall provide an answer regarding the existence of such rights and their applicability in the national legal systems. Second, it shall verify the relation between the directive and the national implementation: it will identify where differences lie and highlight those cases in which the departure from the directive entails a violation of it.

In this paper, we shall present an experiment made with Arg2P, a logic-based argumentation tool built on top of Prolog, which enables defeasible reasoning and argumentation and deals with priorities over rules. Prolog theories developed in Crossjustice are called and executed by Arg2P as modules.

3. The technology: Arg2P

Arg-tuProlog (Arg2P in short) [8, 9] is a lightweight Prolog-based implementation for structured argumentation [10] in compliance with the micro-intelligence definition [11, 12].

According to the ASPIC$^+$ [13] model, Arg2P includes both strict and defeasible rules. Arg2P offers a twofold syntax: Prolog-like, where conclusions precede premises, or ASPIC$^+$-like syntax, where premises precede conclusions in the rule syntax[2].

Strict rules are represented in the Prolog-like syntax form

$$c :- p_1, \ldots, p_n$$

or in the equivalent ASPIC-like syntax:

$$label : p_1, \ldots, p_n -> c$$

where where $label$ is the identifier of a rule and conclusion c, as well as premises $p_1, \ldots, p_n$, are Prolog terms.

Defeasible rules are represented in the form

$$c :\Rightarrow p_1, \ldots, p_n$$

or in the equivalent ASPIC-like syntax:

$$label : p_1, \ldots, p_n => c$$

Arg2P allows two type of negation to be used:

- $-$, to indicate a strong negation (contrary), that captures a notion of negation as definite falsity;
- $\backslash+$ (Prolog-like) or $\sim$ (ASPIC-like) to indicate weak negation (negation as failure) in rule premises.

[2]The Prolog-like syntax is disabled by default and can be enabled via the *prologStrictCompatibility* flag

The ASPIC$^+$ formalism has been extended to capture some patterns of legal reasoning—such as, the burden of persuasion [14, 15, 16] and deontic concepts [17], i.e., explicit representation of obligations and permissions.

The Arg2P model also allows for different semantics to be computed using the *argumentLabellingMode* flag that can assume the values *grounded*, *complete* (both detailed in [18]), *bp_grounded_partial* and *bp_grounded_complete* (both detailed in [14]). Note that additional flags can be used during the resolution process. For a detailed overview please refer to the Arg2P home page[3].

3.1. Modular Argumentation

Arg2P supports modular argumentation [19] that enables reasoning with rules and interpretations of multiple legal systems. This feature makes it possible to design and define knowledge organised in distinct and separate modules that can "call" one another. In particular, a knowledge module – which may represent legal systems or parts of it – can be used by itself, or by referring to another module for specific issues. This second approach is done by directly calling and querying the relevant module.

For instance, in the judicial domain, the application and enforcement of the law make it necessary to take into account the interplay of multiple normative systems, especially when dealing with international contracts and other commercial and social interactions involving different countries. Moreover, normative systems may also interact or conflict on different levels: this is true of both national legal systems and of various transnational or international laws and conventions. All these sources of law need to be considered to properly reason about the law and can be modelled in the Arg2P tool as distinct modules.

Modules are identified by distinct Prolog files (`.pl` files) and can be called and executed using the predicate `module_call(+Modules, :Query)`, where `Modules` is an input parameter containing the list of the required modules – i.e., modules that need to be loaded to answer the query – and `Query` is the query that must be evaluated. In particular, the predicate: *i)* creates a new environment that contains *only* the required modules data, *ii)*, executes the query in the newly created environment, *iii)* and feeds the result to the caller-—note that the original caller environment is not altered by the procedure.

4. Example and discussion

In the following section, two focus cases will be analysed. The first shall demonstrate how the Arg2P tool – merging logic programming and argumentation capabilities – can highlight an explicit contrast between EU and national laws, concerning, in particular, a definition given by the Polish legislator which is different from those provided by EU law and other national laws. The second will provide the user a visual representation of the conformity between the national legislation and the European Directive.

Example 1 (Case 1). *First, let us introduce the premises of the focus case. For simplicity and clarity's sake, we will only take into consideration the rules which are of interest to us and ignore*

[3] http://arg2p.apice.unibo.it

any other legal issue concerning the applicability of the modelled norms. The purpose of this paper is to emphasise how argumentation can provide an explainable answer. One of the points of contention between the Polish national law and the European Directives concerns the notion of a 'child'. Article 3 paragraph 1 of the Directive 2016/800 states that child means a person below the age of 18, and thus the directive shall apply to all persons who committed a crime before that age, with some exceptions.

*In this context, the word that Polish criminal law system usually uses is minor (*nieletni*). This is legally defined in a separate Act on Proceedings in Juvenile Cases as a person who engaged in a punishable act being at least 13 but no more than 17 years old at the moment of the deed. This notion thus does not cover persons who engaged in criminal activity after turning 17 years old, although one has to bear in mind that there are certain provisions that grant some right to persons below 18 years old but are not minors.*

In our example, we shall then try and verify whether Article 15 of the directive – which states that children have the right to be accompanied by the holder of parental responsibility – applies to individuals being 17 years old. We shall also compare the provision in the directive to its Polish transposition, namely Article 32, letter f, of the Act on Proceedings in Juvenile Cases [hereinafter Act], which guarantees the same right to minors.

We shall thus assume that a person has engaged in criminal activity, punishable by law, in Poland, while he or she was 17. After the investigation, the person was therefore formally charged with a crime, while still in Poland, and subsequently summoned to court for the hearing. The defendant now decides to interrogate the decision-support system for his or her rights in that instance.

Example 2 (Case 2). *In this example, we shall then try and verify whether Article 16 of the Directive applies, which states that children have the right to be present at their trial. The Polish legislator has not directly transposed this article, stating that the general right of accused persons (regardless of age), to participate in the main trial is enough to guarantee the applicability of that right, as stated in Article 374 of the Code of Criminal Procedure. Specific rights for juveniles deprived of liberty do exist, although they are under more stringent requirements, such as the necessity for the correct exercise of his right to defence and the submission of a request to that effect, as stated in Article 62, paragraph 1, of the Act.*

We shall thus assume that a person has engaged in criminal activity, punishable by law, in Poland, while he or she was under 17 years of age. After the investigation, the person was then formally charged with a crime, while still in Poland, placed in a temporary detention centre for juveniles. The defendant subsequently requests to be summoned to court to explain his actions and decides to interrogate the decision-support system for his or her rights in that instance.

4.1. Incorrect Transposition: diverging implementation

Listing 1: Rules enconding for Example 1

```
% Directive

% rule_8
has_right(Article, PersonId, Right, Matter, directive_2016_800) :-
    directive_applies(PersonId),
    has_right(Article, PersonId, Right, Matter).
```

```
7
8   % rule_7
9   directive_applies(PersonId) :-
10    person_status(PersonId, child).
11
12  % rule_6
13  person_status(PersonId, child) :-
14      user_fact(person_age(PersonId, X)),
15      X < 18.
16
17  person_status(PersonId, adult) :-
18      user_fact(person_age(PersonId, X)),
19      X >= 18.
20
21  % rule_9
22  has_right(article15_1, PersonId, right_to_be_accompanied, HolderId) :-
23      user_fact(person_status(HolderId, holder_of_parental_responsibility)),
24      user_fact(proceeding_status(PersonId, started)),
25      user_fact(proceeding_matter(PersonId, court_hearing)).
26
27  % Polish implementation
28
29  law1982_applies(article1_2, PersonId) :=
30      user_fact(proceeding_type(PersonId, criminal)),
31      user_fact(person_age(PersonId, X)),
32      X > 13,
33      X < 17.
34
35  % rule_5
36  person_status(PersonId, adult) :=
37      user_fact(person_age(PersonId, X)),
38      X >= 17.
39
40  person_status(PersonId, minor) :=
41      law1982_applies(article1_1, PersonId).
42
43  has_right(article32f, PersonId, right_to_be_accompanied, HolderId) :=
44      person_status(PersonId, minor),
45      user_fact(person_status(HolderId, holder_of_parental_responsibility)).
46
47  % Facts
48
49  user_fact(person_age(nino, 17)).                                    % rule_1
50  user_fact(person_status(alf, holder_of_parental_responsibility)).   % rule_2
51  user_fact(proceeding_matter(nino, court_hearing)).                  % rule_3
52  user_fact(proceeding_status(nino, started)).                        % rule_4
53
54  % Conflict
55
56  conflict([person_status(PersonId, child)], [person_status(PersonId, adult)]).
57  conflict([person_status(PersonId, adult)], [person_status(PersonId, child)]).
```

Listing 1 illustrates the rules that apply in the first focus case. The first five rules, listed under the commented label 'Directive', show the modelling of articles pertaining to the Directive 2016/800, while the ones under the label 'Polish implementation' are the transposition of articles of the Polish Code of Criminal Procedure and the Act. Under the label 'Facts' are the elements that the user needs to input, and are added accordingly to the factual situation and the object of the query. Lastly, the label 'Conflict' illustrates the target result of the use of argumentation. It describes the conflictual relationship that exists between the two definitions of child and of adult.

We shall now proceed to the explanation of the meaning of each rule, and its impact on the result of the focus case. The goal the first rule tries to prove, namely the fact that the right to be accompanied exists, depends on two premises: the first premise states that the Directive (2016/800) shall apply, when connected to a certain person, while the second states that the person to whom the directive applies shall have any of the rights as stated in the Directive, according to a specific article, and that right may be present with an additional attribute or characteristic (that is the argument `Matter`), such as the phase of the trial or the authority responsible for the right.

Following the first rule, we see that the directive applies when the person has been given the status of child (there are many articles describing the scope of the directive but for the purpose of this example we shall only take one into consideration). According to the second rule, a person is a child when his age is lower than 18. The third rule, not found in the directive but clearly implied by it, states that a person is an adult when his age is equal to or higher than 18. The fourth rule says that a person has the right to be accompanied to the court hearing when the person is a child.

After the rules modelling the Directive, we have the rules modelling the Polish transposition. The first of such rules which that a minor has a right to be accompanied by a holder of parental responsibility, identified by the `HolderId`, according to article15(1), whenever proceedings have started and the stage of concern is the court hearing.

The first rule under the label 'Polish implementation' states that the Act shall apply only when the proceedings concern a criminal matter and the person is between the ages of 13 and 17. Therefore we can infer that a person shall be tried as an adult whenever he is aged 17 and above.

The final rule of this example consists in the representation of article 32f of the Act, which, for the purpose of our knowledge-base, we have transposed as stating that whenever a person is a minor, and the holder of parental responsibility has been identified, the minor shall have the right to be accompanied by him.

It is necessary to remark that all rules in the Polish implementation are defeasible, while those in the directive are strict.

By modelling the EU rules and strict and the national rules as defesible, we give preference to the first over the second. In this way, in our current implementation, we can easily visualise whenever the latter is in violation of the European directive, although this preference structure can be changed to better fit our scope and purpose.

The facts that we input into the system state that a person, called `nino`, is 17 years of age, and a person, named `alf`, is `nino`'s holder of parental responsibility. Furthermore, we state that `nino`'s proceedings have started and that the instance of concern to us is the court hearing.

The final two lines of code transpose the conflict that exists between the definitions of child and adult. Whenever a person is a child, the system shall infer that a person cannot be an adult, and the opposite applies accordingly.

```
A0 : rule_1 ⟹ user_fact(person_age(nino, 17))
A1 : rule_2 ⟹ user_fact(person_status(alf, holder_of_parental_responsibility))
A2 : rule_3 ⟹ user_fact(proceeding_matter(nino, court_hearing))
A3 : rule_4 ⟹ user_fact(proceeding_status(nino, started))
A4 : A0,rule_5 ⟹ person_status(nino, adult)
A5 : A0,rule_6 ⟹ person_status(nino, child)
A6 : A5,rule_7 ⟹ directive_applies(nino)
A7 : A6,A8,rule_8 ⟹ has_right(article15_1, nino,
    right_to_be_accompanied, alf, directive_2016_800)
A8 : A1,A3,A2,rule_9 ⟹ has_right(article15_1, nino,
    right_to_be_accompanied, alf)
```

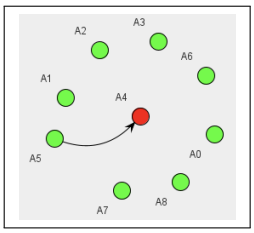

Figure 1: Arguments for Example 1 (left) and corresponding graph (right)

Figure 1 shows the generated arguments (left) and the results of the framework evaluation according to grounded semantic (right). The system returns the resulting applicable right from the facts added as input. Therefore, in our focus case, the defendant does indeed have the

right to be accompanied by the holder of parental responsibility, according to Article 15 of the Directive 2016/800 (argument A8), but the same right is not granted by the Polish law.

While backtracking through each individual rule and finding the difference is possible in a case such as ours, due to the extremely limited number of rules, in a real-life scenario it would result in having to look through the entire rule-set, thus through the entire code.

The added inclusion of argumentation to our expert system illustrates visually the attack-defeat relationship that occurs between the definitions of child and adult. As the person is simultaneously a child, according to the directive, and an adult, according to the Polish law, the two facts that are inferred from the aforementioned rules are in conflict with each other, thus the arrows leading from A5 to A4 illustrate the direct conflict that occurs. The European definition defeats the Polish one, due to the defeasible nature of the latter [4].

The information given to the user is not limited to the final goal that has been satisfied by the system. Argumentation provides any person interested in understanding the origin of the conflict with the exact clause that is the source of the differences between the two legal sources. In our case, the user would be able to verify that the conflict between EU and Polish law, derives from a different definition of the EU status of child and the Polish status of minor.

While this example shows the usefulnees of argumentation in highlighting explicit contrasts in the rule-base, all conflicts have to preemptively be added into the system by the knowledge engineer. The system is thus able to explain its result only as long as the explanation is already present in the rule-base itself.

4.2. Incorrect Transposition: diverging requirements

For the second focus case, we shall illustrate the use of argumentation to verify the conformity between directives and national laws. The conflict, for this second example, is created by comparing the set of facts created by the rules of the Directive and the set created by the Polish national transposition. The system shall thus verify whether the facts in the two sets are equal. The variables `F` and `L` – `generate` rule (Listing 2) – stand for facts and length respectively and are instantiated by the creation of a set of facts that are needed to have the goal succeed.

The new rules – the `directiveconformity` rule and the dual one `directivenotconformity` – shall check whether both the directive and the Polish national laws have returned at least one result with the same `Right`, applicable to the same `Person`.

Listing 2: Rules enconding for Example 2

```
modulesPath('home/crossjustice-explainability').

generate :
    module(Module),
    prolog(call_module([Module, 'facts'], with_facts_and_length(has_right(X, Y, Z, U, S), F, L)))
        => right(Module, X, Y, Z, U, F, L).

directiveconformity :
  right(directive, X, PersonId, Right, U, F, L),
  right(polish, XX, PersonId, Right, UU, FF, LL)
    => conformity(polish, PersonId, Right).

directivenotconformity :
```

[4] If both the Polish law and the directive were to be transposed as defeasible norms, no argument would defeat the other, and the resulting conflict would be unresolved. If both were to be transposed as strict, no conflict can arise.

```
  right(directive, X, PersonId, Right, U, F, L),
  ~(right(polish, XX, PersonId, Right, UU, FF, LL))
    => -conformity(polish, PersonId, Right).

module1 :-> module('directive').
module2 :-> module('polish').

conflict([right(directive, XX, A, Z, U, F, L)], [right(polish, X, A, Z, U, FF, LL)]) :-
    \+ conflictFunction(F, FF).
conflict([right(polish, XX, A, Z, U, F, L)], [right(directive, X, A, Z, U, FF, LL)]) :-
    \+ conflictFunction(FF, F).

conflictFunction(F, FF) :-
    sameFacts(FF, F),
    sameFacts(F, FF).

sameFacts([], _).
sameFacts([H|T], Facts) :-
    member(H, Facts),
    sameFacts(T, Facts).
```

Listing 3: Rules enconding for Example 2 - Directive Articles

```
has_right(Article, PersonId, Right, Matter, directive_2016_800) :-
    person_status(PersonId, child),
    has_right(Article, PersonId, Right, Matter).

person_status(PersonId, child) :-
    user_fact(person_age(PersonId, X)),
    X < 18.

has_right(article6_1, PersonId, right_to_access_lawyer, trial) :-
    user_fact(person_status(PersonId, suspect)),
    user_fact(proceeding_type(PersonId, criminal)).

has_right(article6_1, PersonId, right_to_access_lawyer, trial) :-
    user_fact(person_status(PersonId, accused)),
    user_fact(proceeding_type(PersonId, criminal)).

has_right(article16_1, PersonId, right_to_be_present, trial) :-
    person_status(PersonId, child).
```

Listing 3 shows a list of articles found in the directive. The structure of the ontology is the same as the example above. We have added for this focus case the transposition of article 6 paragraph 1 and article 16 paragraph 1. For the right to access a lawyer to be recognised, the Directive states that a person must be a suspect or accused, the proceeding must be related to criminal matters and the person must be a child. These are the preconditions that make up the set related to the first right.

For the right to be present at the trial to be recognised, the Directive only states that the person must be a child. This is the only precondition that makes up the set related to the second right.

Listing 4: Rules enconding for Example 2 - Polish national laws articles

```
has_right(Article, PersonId, Right, Matter, directive_2016_800_polish) :-
    has_right(Article, PersonId, Right, Matter).

criminal_code_applies(article10_1, PersonId) :-
    user_fact(person_age(PersonId, X)),
    X >= 17.

law1982_applies(article1_2, PersonId) :-
    user_fact(proceeding_type(PersonId, criminal)),
    user_fact(person_age(PersonId, X)),
    X > 13,
    X < 17.

person_status(PersonId, minor) :-
    law1982_applies(article1_2, PersonId).

has_right(article79_1_1, PersonId, right_to_access_lawyer, trial) :-
    user_fact(person_status(PersonId, accused)),
```

```
    user_fact(proceeding_type(PersonId, criminal)),
    user_fact(person_age(PersonId, X)),
    X < 18.

has_right(article62_1, PersonId, right_to_be_present, trial) :-
    person_status(PersonId, minor),
    user_fact(person_status(PersonId, temporarily_detained)),
    user_fact(person_request_submitted(PersonId, present_trial)).

has_right(article374_1, PersonId, right_to_be_present, trial) :-
    user_fact(person_status(PersonId, accused)).
```

Listing 4 shows a list of articles found in the Polish national laws. The structure of the ontology is the same as Example 1. We have added for this focus case the transposition of article 79 paragraph 1, article 62 paragraph 1, and article 374 paragraph 1. In order for the right to access a lawyer to be successfully recognised, the Polish code of Criminal Procedure states that a person must be a suspect or accused, the proceeding must be related to criminal matters and the person must be under 18 years of age. These are the three elements that make up the set related to the first right.

For the right to be present at the trial to be recognised, the Polish legislator has two articles that are meant to transpose of the European directive. Article 374 of the code of Criminal Procedure only states that a person must be accused, making this the only precondition that makes up the set related to the second right.

Additionally, article 62 also exists, which states that for temporarily detained minors (thus persons under 17 years of age), there must be the additional precondition of a request submitted by the defendant.

Listing 5: Rules enconding for Example 2

```
user_fact(person_age(nino, 16)).
user_fact(proceeding_type(nino, criminal)).
user_fact(person_status(nino, temporarily_detained)).
user_fact(person_request_submitted(nino, present_trial)).
user_fact(person_status(nino, accused)).
```

The facts that we input into the system state that a person, called `nino`, is 16 years of age. Furthermore, we state that `nino`'s proceedings are of a criminal nature and that `nino` has been placed into temporary detention, becoming accused after he has been formally charged with a crime. We also state that `nino` has submitted a request to be present at his trial (Listing 5).

Figure 2 shows the generated arguments (left) and the results of the framework evaluation according to grounded semantic (right). The system returns the resulting applicable right from the facts we have added as input.

Note that in this case the Prolog-like syntax evaluation of Arg2P has not been used. Instead, the Prolog code in the modules is evaluated through the rule `generate`. The evaluation of the Prolog goal `has_right` is done thanks to the special Arg2P predicate `prolog/1` allowing the evaluation of pure Prolog code inside the ASPIC-like syntax. The results obtained from the deductive reasoning are then exploited to build the `right` arguments. Argumentation is here used as an upper layer on the already existing logical system delivering a higher level of explainability.

With regard to the right to access legal aid, we can see that both the directive (argument A3) and the Polish law (argument A6) have returned the same right, with the same conditions. From the argumentation framework, we can then verify that the Polish law is a perfect transposition

```
A0 : module1 ⟹ module(directive)
A1 : module2 ⟹ module(polish)
A2 : A0,generate ⟹ right(directive, article16_1, nino, right_to_be_present, trial,
        [person_age(nino, 16),
        person_age(nino, 16)], 2)
A3 : A0,generate ⟹ right(directive, article6_1, nino, right_to_access_lawyer, trial,
        [proceeding_type(nino, criminal),
        person_status(nino, accused),
        person_age(nino, 16)], 3)
A4 : A1,generate ⟹ right(polish, article374_1, nino, right_to_be_present, trial, [
        person_status(nino, accused)], 1)
A5 : A1,generate ⟹ right(polish, article62_1, nino, right_to_be_present, trial, [
        person_request_submitted(nino, present_trial),
        person_status(nino, temporarily_detained),
        person_age(nino, 16),
        proceeding_type(nino, criminal)], 4)
A6 : A1,generate ⟹ right(polish, article79_1_1, nino, right_to_access_lawyer, trial,
        [person_age(nino, 16),
        proceeding_type(nino, criminal),
        person_status(nino, accused)], 3)
A7 : A3,directivenotconformity ⟹ -conformity(polish, nino, right_to_access_lawyer)
A8 : A2,directivenotconformity ⟹ -conformity(polish, nino, right_to_be_present)
A9 : A3,A6,directiveconformity ⟹ conformity(polish, nino, right_to_access_lawyer)
A10 : A2,A4,directiveconformity ⟹ conformity(polish, nino, right_to_be_present)
```

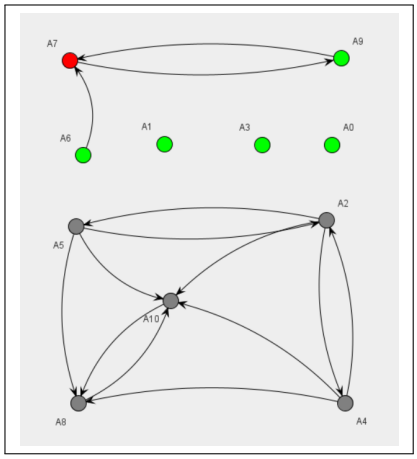

Figure 2: Arguments from Example 2 (left) and corresponding graph (right)

of the European directive, as the argument for the conformity to the directive (A9) attacks and defeats its negative counterpart (A7), due to the presence of the Polish right (A6).[5]

With regard to the right to be present at the trial the situation is different. The right is present once according to the directive (A2), and twice according to the Polish law (A4 and A5), although neither of the two national implementations contains the same arguments as the European source. On the one hand, the system finds that the same right is returned by both the European and Polish modules, on the other hand, neither implementation contains the same requirements for the right to be guaranteed, therefore the system cannot decide whether the Polish law has successfully implemented the directive.

The graph shows that the European right (A2) is in conflict with each of the Polish rights (A4 and A5), and that all three arguments attack the argument for conformity (A10), as neither is a complete transposition. Both Polish rights also attack the argument for the `notConformity`, as they are indeed returning the same right as the directive. The system thus cannot conclude in one way or the other, and the arguments remain greyed out.

The user would therefore be provided with the information that, although the right exists in the Polish legal system, we cannot say the right has been fully implemented in the national legal system, and an issue of applicability shall arise. Highlighting such contrasts can provide a better understanding of any underlining legal concerns that a traditional expert system cannot easily provide.

To conclude the analysis of this focus case, we must remark upon the next step for the development of our system. First, any conflict that arises in the framework carries its attack/defeat relation through the subsequent inferences that within an argumentation framework. To illustrate the significance of argumentation we have limited the number of applicable rules in our focus cases, but if we were to have longer inference paths and multiple rules that function as both premise and conclusion, we could demonstrate how, if a premise were found to have

[5]If the Polish law did not have an article implementing the right, the result would be the opposite, as the argument `notConformity` would be the only one present, due to the absence of the Polish right.

been defeated, any inference built upon that argument would also be defeated.

Both conflict relations we have presented in this paper – a conflict on the outcome and a conflict on the requirements – can be elicited and presented to the users. The system thus can provide users both the source of the conflict, as well as a list of facts that are needed to create the conflict. By analysing the set of factors required and the outcome of the system, it is possible to restrict the number of rules one has to look through to find the premise(s) that are in direct opposition, i.e., that trigger incompatible consequences through the rules.

4.3. Conclusion

The paper exemplifies how logic programming and argumentation can help to obtain explainable intelligent systems.

The proposed solution is based on a lightweight argumentation technology based on logic programming—namely Arg2P. A full example is presented discussing its application in a real legal scenario—the CrossJustice project. The discussion and the corresponding example already highlight the potential benefits of the approach. In particular, the discussion carried out in the paper in the CrossJustice case points out the advantages of using argumentation in combination with a knowledge based modelled using logic programming techniques.

The results presented here represent just a preliminary exploration of the intersection between logic programming and explainability, but it has the potential to work as a starting point for further research.

Acknowledgments

This work has been partially supported by the H2020 ERC Project "CompuLaw" (G.A. 833647) and by the European Union's Justice Programme (Grant Agreement No. 847346) for the project "Knowledge, Advisory and Capacity Building Information Tool for Criminal Procedural Rights in Judicial Cooperation".

References

[1] R. Calegari, G. Ciatto, E. Denti, A. Omicini, Logic-based technologies for intelligent systems: State of the art and perspectives, Information 11 (2020) 1–29. doi:10.3390/info11030167.

[2] R. Calegari, G. Ciatto, A. Omicini, On the integration of symbolic and sub-symbolic techniques for XAI: A survey, Intelligenza Artificiale 14 (2020) 7–32. URL: http://content.iospress.com/articles/intelligenza-artificiale/ia190036. doi:10.3233/IA-190036.

[3] H. Prakken, R. Ratsma, A top-level model of case-based argumentation for explanation: Formalisation and experiments, Argument & Computation (2021) 1–36.

[4] R. Calegari, A. Omicini, G. Sartor, Argumentation and logic programming for explainable and ethical AI, in: C. Musto, D. Magazzeni, S. Ruggieri, G. Semeraro (Eds.), XAI.it 2020 – Italian Workshop on Explainable Artificial Intelligence 2020, volume 2742 of *CEUR Workshop Proceedings*, Sun SITE Central Europe, RWTH Aachen University, 2020, pp.

55–68. URL: http://ceur-ws.org/Vol-2742/paper5.pdf, proceedings of the Italian Workshop on Explainable Artificial Intelligence co-located with 19th International Conference of the Italian Association for Artificial Intelligence (AIxIA 2020).

[5] R. Calegari, G. Ciatto, V. Mascardi, A. Omicini, Logic-based technologies for multi-agent systems: A systematic literature review, Autonomous Agents and Multi-Agent Systems 35 (2021) 1:1–1:67. URL: http://link.springer.com/10.1007/s10458-020-09478-3. doi:10.1007/s10458-020-09478-3, collection "Current Trends in Research on Software Agents and Agent-Based Software Development".

[6] R. Calegari, A. Omicini, G. Sartor, Explainable and ethical AI: A perspective on argumentation and logic programming, in: M. Baldoni, S. Bandini (Eds.), AIxIA 2020 – Advances in Artificial Intelligence, volume 12414 of *Lecture Notes in Computer Science*, Springer Nature, 2021, pp. 19–36. URL: http://link.springer.com/10.1007/978-3-030-77091-4_2. doi:10.1007/978-3-030-73065-9_2.

[7] K. D. Ashley, A brief history of the changing roles of case prediction in ai and law, Law Context: A Socio-Legal J. 36 (2019) 93.

[8] G. Pisano, R. Calegari, A. Omicini, G. Sartor, Arg-tuprolog: a tuprolog-based argumentation framework, in: Proceedings of the 35th Italian Conference on Computational Logic - CILC 2020, volume 2710 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020, pp. 51–66. URL: http://ceur-ws.org/Vol-2710/paper4.pdf.

[9] R. Calegari, G. Contissa, G. Pisano, G. Sartor, G. Sartor, Arg-tuProlog: a modular logic argumentation tool for PIL, in: S. Villata, J. Harašta, P. Křemen (Eds.), Legal Knowledge and Information Systems. JURIX 2020: The Thirty-third Annual Conference, volume 334 of *Frontiers in Artificial Intelligence and Applications*, 2020, pp. 265–268. doi:10.3233/FAIA200880.

[10] P. Besnard, A. J. García, A. Hunter, S. Modgil, H. Prakken, G. R. Simari, F. Toni, Introduction to structured argumentation, Argument & Computation 5 (2014) 1–4. doi:10.1080/19462166.2013.869764.

[11] A. Omicini, R. Calegari, Injecting (micro)intelligence in the IoT: Logic-based approaches for (M)MAS, in: D. Lin, T. Ishida, F. Zambonelli, I. Noda (Eds.), Massively Multi-Agent Systems II, volume 11422 of *LNCS*, Springer, 2019, pp. 21–35. doi:10.1007/978-3-030-20937-7{\char95}2.

[12] R. Calegari, Micro-Intelligence for the IoT: Logic-based Models and Technologies, Ph.D. thesis, Alma Mater Studiorum—Università di Bologna, Bologna, Italy, 2018. URL: http://amsdottorato.unibo.it/8521/.

[13] S. Modgil, H. Prakken, The aspic+ framework for structured argumentation: a tutorial, Argument & Computation 5 (2014) 31–62.

[14] R. Calegari, G. Sartor, Burden of persuasion in argumentation, in: F. Ricca, A. Russo, S. Greco, N. Leone, A. Artikis, G. Friedrich, P. Fodor, A. Kimmig, F. Lisi, M. Maratea, A. Mileo, F. Riguzzi (Eds.), Proceedings 36th International Conference on Logic Programming (Technical Communications), volume 325 of *Electronic Proceedings in Theoretical Computer Science*, Open Publishing Association, 2020, pp. 151–163. doi:10.4204/EPTCS.325.21.

[15] R. Calegari, R. Regis, G. Sartor, The burden of persuasion in structured argumentation, in: J. Maranhão, A. Z. Wyner (Eds.), ICAIL'21: Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law, ICAIL'21, ACM, 2021, p. 180–184. URL:

https://dl.acm.org/doi/10.1145/3462757.3466078. doi:10.1145/3462757.3466078.
[16] R. Calegari, G. Sartor, A model for the burden of persuasion in argumentation, in: S. Villata, J. Harašta, P. Křemen (Eds.), Legal Knowledge and Information Systems. JURIX 2020: The Thirty-third Annual Conference, volume 334 of *Frontiers in Artificial Intelligence and Applications*, 2020, pp. 13–22. doi:10.3233/FAIA200845.
[17] R. Riveret, A. Rotolo, G. Sartor, A deontic argumentation framework towards doctrine reification, Journal of Applied Logics—IfCoLog Journal of Logics and their Applications 6 (2019) 903–940. URL: https://collegepublications.co.uk/ifcolog/?00034.
[18] P. Baroni, M. Caminada, M. Giacomin, An introduction to argumentation semantics, The Knowledge Engineering Review 26 (2011) 365–410. doi:10.1017/S0269888911000166.
[19] R. Calegari, G. Contissa, G. Pisano, G. Sartor, G. Sartor, Arg-tuProlog: a modular logic argumentation tool for PIL, in: S. Villata, J. Harašta, P. Křemen (Eds.), Legal Knowledge and Information Systems. JURIX 2020: The Thirty-third Annual Conference, volume 334 of *Frontiers in Artificial Intelligence and Applications*, 2020, pp. 265–268. doi:10.3233/FAIA200880.