Automated Legal Reasoning with Discretion to Act using s(LAW)

Joaquín Arias Mar Moreno-Rebato
Jose A. Rodriguez-García Sascha Ossowski

CETINIA, Universidad Rey Juan Carlos, Madrid, 28933, Spain.

Abstract

Automated legal reasoning and its application in smart contracts and automated decisions are increasingly attracting interest. In this context, ethical and legal concerns make it necessary for automated reasoners to *justify* in human-understandable terms the advice given. Logic Programming, specially Answer Set Programming, has a rich semantics and has been used to very concisely express complex knowledge. However, modelling *discretionality to act* and other vague concepts such as *ambiguity* cannot be expressed in top-down execution models based on Prolog, and in bottom-up execution models based on ASP the justifications are incomplete and/or not scalable. We propose to use s(CASP), a top-down execution model for predicate ASP, to model vague concepts following a set of patterns. We have implemented a framework, called s(LAW), to model, reason, and justify the applicable legislation and validate it by translating (and benchmarking) a representative use case, the criteria for the admission of students in the "Comunidad de Madrid".

Keywords: Answer Set Programming, Goal-Directed, Ambiguity, Administrative Discretion.

This work has been supported by grant VAE: TED2021-131295B-C33 funded by MCIN/AEI/ 10.13039/501100011033 and by the "European Union NextGenerationEU/PRTR", by grant COSASS: PID2021-123673OB-C32 funded by MCIN/AEI/ 10.13039/501100011033 and by "ERDF A way of making Europe", and by grant 2023/00004/004 s(LAW) funded by URJC.

1 Introduction

The formal representation of legal texts to automatize reasoning about them is well known in literature. For deterministic rules there are several proposals, often based on logic programming languages (Ramakrishna et al, 2016; Sergot et al, 1986). This topic is recently gaining much attention thanks to the interest in the so-called smart contracts, and to automated decisions by public administrations (Cerrillo i Martínez, 2020; Cobbe, 2019; Huergo Lora, 2020; Suksi, 2021; Vestri, 2021).

Law uses a natural language that is characterized by its vagueness, ambiguity, and open texture (which admits both restrictive and extensive interpretations), i.e., there are areas of certainty and areas of uncertainty. This contrasts with other scientific disciplines built on the use of symbolic or mathematical languages. Modeling legal rules with computer languages is, therefore, a complex task, since indeterminate legal concepts, discretionary powers, general principles of law, etc., need to be formally represented. Perhaps counterintuitively, even in the case of *regulated* procedures the formal modeling of legal rules may raise multiple problems, because in most cases it will be necessary to interpret the legal rule (specifying indeterminate legal concepts, applying general principles of law, connecting some legal rules with others, etc.). The modeling of legal rules in procedures in which *discretionary* powers are articulated, increases this complexity significantly. To meet the challenge of automating administrative procedures, it is necessary to duly represent the aforementioned concepts pertinent to legal language within a computer-interpretable formal language.

However, existing proposals usually fall short in adequately capturing the ambiguity and/or administrative discretion present in applicable legislation. A good example is *force majeure*. Force majeure is a law term that must be understood as referring to abnormal and unforeseeable circumstances which were outside the control of the party by whom it is pleaded and the consequences of which could not have been avoided in spite of the exercise of all due care (see judgment Court of Justice of European Union, case Tomas Vilkas, C-640/15, 25 January 2017). Consider, for instance, the awarding school places in centers supported with public funds in the "Comunidad de Madrid" (CM), in Spain. In the corresponding legal procedure, the proximity of a school to a family's home or work address plays an important role. This proximity is determined based on existing educational districts, except in cases of force majeure, but these cases are not defined a priory. Educational districts which in the current regulations of the Community of Madrid is the municipality as a whole or each of the city districts in the municipality of Madrid.

In this work we present a framework, called s(LAW), that allows for modeling legal rules involving ambiguity, and supports reasoning and inferring conclusions based on them. Additionally, thanks to the goal-directed execution of s(CASP), the underlying system used to implement our proposal, s(LAW) provides justification of the resulting conclusions (in natural language).

To evaluate the expressiveness of our proposal we have represented the criteria for the admission of students in the “Comunidad de Madrid” in s(LAW). In particular we focus on the procedure for awarding school places for the “Educación Secundaria Obligatoria” (ESO) of centers supported with public funds in the CM. The Spanish Organic Law on Education[1] regulates, in article 84, the criteria for the admission of students in public centers and private subsidized centers and, in its second paragraph of this article 84, indicates adjudication criteria. However, since Spain is a politically decentralized country, it is the autonomous communities (and, therefore, their educational administrations) that have powers to develop these aspects of basic state legislation. The CM, in use of its powers in educational matters, establishes the framework and general procedure for the admission of students to educational centers supported with public funds for the ESO.[2] The case presented in this paper is, therefore, real case, based on the regulations currently in force.

The present article is structured as follows. Section 2 provides a brief description of the field of goal-directed Answer Set Programming that the s(LAW) framework sets out from. Section 3 analyses the legal basis of Discretion to Act using Spanish legislation as an example. In section 4 we show how Discretion to Act, and other relevant concepts, can be modelled withing the s(LAW) automated reasoner, using the awarding of school places for the ESO as a running example. In Section 5 we describe how s(LAW) can generate natural language *explanations* for its conclusions drawn from legal reasoning with Discretion to Act. Section 6 discusses related work while in Section 7 we point to future lines of work.

2 Goal-Directed Answer Set Programming

Our proposal relies on Answer Set Programming (ASP) (Gelfond and Lifschitz, 1988) for coding legal rules. More specifically, we use s(CASP) (Arias et al, 2018), a goal-directed implementation of ASP that features predicates, constraints among non-ground variables, and uninterpreted functions.

The top-down query-driven execution strategy of s(CASP) has three major advantages w.r.t. traditional ASP system: (a) it does not require to ground the programs; (b) its execution starts with a query and the evaluation only explores the parts of the knowledge base relevant to the query; and (c) s(CASP) returns partial stable models (the relevant subsets of the ASP stable models

[1] Organic Law 2/2006, May 3, last modified by Organic Law 3/2020, December 29

[2] Decree 29/2013, of 11 April, amended by Decree 244/2021, of 29 December, of the “Consejo de Gobierno”, on freedom of choice of school in the “Comunidad de Madrid” and updating the admission criteria and their weighting; Order 1240/2013, of 17 April, of the “Departamento de Educación, Juventud y Deportes” of the “Comunidad de Madrid”, amended by Order 1534/2019, of May 17 and by Order 592/2022, of 18 March, of the “Consejería de Educación e Investigación” of the “Comunidad de Madrid”; Resolution of July 31, 2013, of the “Dirección General para la Mejora de la Calidad de la Educación” (in relation to bilingual education); and Joint Resolution of the “Viceconsejería de política educativa” and “Viceconsejería organización educativa” by which instructions are issued to carry out the actions prior to the process of admission of students in centers supported with public funds for the 2022/2023 academic year, of November 25, 2022. (https://www.comunidad.madrid/sites/default/files/doc/educacion/resolucion_conjunta_admision_regimen_general_2023-2024_.pdf).

needed to support the query) and their corresponding justification (proof tree). Thus, our proposal automates commonsense reasoning and is scalable whereas ground-based ASP systems do not (Section 6).

Additionally, s(CASP) provides a mechanism to present justifications in natural language using a generic translation, and the possibility of customizing them with directives that provide explanation patterns in natural language (Arias et al, 2020). Both plain text and user-friendly, expandable HTML can be generated. These patterns can be used with the program text itself, thereby making it easier for experts without a programming background to understand both the program and the results, i.e., partial model and justification, of its execution.

3 Legal Basis for the Discretion to Act

This section presents the first main contribution of this work, the analysis of the administrative and political discretion to act. In addition, we explain the limitation for automated application of the discretion to act and propose a framework that would facilitate the application of discretion to act.

3.1 Political Discretion and Administrative Discretion

This section gives a brief explanation of when political and/or administrative discretion to act occurs and some examples.

3.1.1 Political Discretion

Political discretionality occurs when, either explicitly or implicitly, political bodies or authorities at the highest political level (governments, ministers, mayors, etc.) or senior officials of the Administration are granted a margin of decision making of their own, choosing between different possible alternatives. In reality, this is politics, the ability to choose in order to pursue the general interest.

This type of discretion is considered a type of strong, or maximum, discretion, inherent to the political responsibility of the person exercising it. Examples of this type of discretion would be, for example, regulatory norms, the implementation of a public service, the decision to connect two cities by train, the adoption of health measures in a pandemic (requirement or not of a passport, curfew, restrictions on mobility or on the hotel industry, etc.).

3.1.2 Administrative Discretion

Concept

Administrative discretionality (Garcia de Enterría, 1962; Garcia de Enterría and Fernández, 2020; Sánchez Morón, 2021; Cosculluela Montaner, 2021; Otenyo, 2016) implies a margin of free choice attributed by a rule to a Public Administration and, within this, to an administrative body. Discretionality implies being able to introduce subjective criteria of valuation, to choose between different alternatives, provided that these are equally lawful. Discretion may refer to the convenience of acting or not, to the manner of acting, or to the content of the action.

Administrative discretion has been contemplated by Huber (1953) as the "Trojan horse of the rule of law". Much theorizing has been done in public law and has emphasized the need to reduce discretionality as much as possible and to fight against the immunities of power, focusing on the scope of its judicial control.

Distinction with respect to regulated acts

An administrative act is a statement issued by a Public Administration and subject to Administrative Law. There are administrative acts that are issued in the exercise of regulated powers and others that are issued in the exercise of discretionary powers.

An administrative act is regulated when the Public Administration cannot introduce any subjective criteria of assessment, i.e., it must simply limit itself to applying the law. Examples from the Spanish administration are three-year terms granted to a civil servant, five-year teaching periods (at the Spanish Universities), or an urban planning license. Let's consider the urban planning license, which is a regulated but complex procedure. Verifying whether a license application complies with all requirements of urban planning and other urban planning regulations is not a simple task, as the applicable legal rules will require a complex legal interpretation. Once the requirements have been met, the Administration – in this case the City Council – must limit itself to granting the license, i.e., it cannot introduce additional requirements or rule out projects that, for example, it dislikes from a political point of view.

On the other hand, there are discretionary acts, i.e., acts that are issued in the exercise of discretionary powers. In these acts, the Public Administration does not limit itself to automatically apply the law. When the Public Administration exercises a discretionary power, it can choose between different alternatives, provided that these are equally valid from the perspective of the Law. Otherwise, it would incur in an arbitrary activity, and arbitrariness is prohibited for public authorities and, therefore, also for the Public Administration (see, for instance, article 9.3. of the "Constitución Española").

Technical discretion

Technical discretion is a subtype of administrative discretion. In this case, the possibility of choosing between different alternatives is limited by the applicable scientific or technical knowledge. Examples would be the academic qualification of an examination or the declaration of ruin of a building on the basis of a technical study. Sometimes it is attributed to collegiate bodies, as may be the case of the evaluation commissions in a competitive process to award teaching positions or research projects.

3.2 Applying discretion to act

An administrative procedure is a set of acts that prepare and enable the last act of the procedure, which is generally a resolution, such as, the granting or refusal of an aid, a subsidy, a research project, a town planning license, or the obligation to pay a fine/tax.

The administrative procedure may be completely regulated, e.g., (i) an application for admission to a public university submitted by someone who meets all requirements and provides the required documentation, or (ii) the granting of an urban planning license, as explained above (although it may seem a simple task, due to the interpretation of legal norms it generally is not).

On the other hand, even for an administrative procedure in which discretionary powers are exercised (those in which the Public Administration can choose between different alternatives, as long as they are equally valid for the Law), discretion is never absolute but subject to legal limits. In every administrative procedure there is always a part (and it is a qualitatively important part) that regulates the action of the Public Administration. Among others, the legal rule always indicates the administrative body that has to exercise that power, the procedure to be followed to adopt the decision, the purpose pursued with the same and the requirements demanded and the verification of the "determining facts" on which the decision is based before exercising the discretion.

As a consequence, an administrative procedure is either entirely regulated, or there will be a regulated part with a small discretionary niche. For instance, in a hypothetical administrative procedure for awarding places in public residences for the elderly, 90 percent of the points to be awarded to the different candidates would be regulated and determined by the income level of the candidates and their socio-health conditions, but remaining 10 percent of points might be awarded by a multidisciplinary team based on criteria that its members would agree upon. Another example is the awarding of school places discussed in the paper. It refers to regulated requirements, with the exception of the small discretionary score that can be awarded by the centers.

3.3 Judicial control/review

The most interesting aspect of the discretion to act is its judicial control (Sourdin (2021)). As we have explained above, discretion means choosing between different alternatives (A, B, C, D...) as long as they are equally valid for the law; that is, as long as they are not arbitrary, discriminatory or illegal because they go against legal norms.

For example, in the context of our running use case, procedure for awarding school places in publicly funded centers:

- Schools may award points for having siblings in the school, or because the parents have been former students, or because the mother has been a victim of gender-based violence and resides near the school,

- but not for belonging to one religion or another, for being of a certain gender, or for having a high IQ.

This configuration of discretion determines that judges cannot enter into judgment on the merits of the discretionary decision. They may supervise the regulated part of the procedure and other elements described below:

- may audit the body exercising jurisdiction,
- whether the procedure has been followed,
- whether the public purpose pursued by the rule has been fulfilled,
- whether the indeterminate legal concepts (those that cannot be configured a priori and that, case by case, must be analyzed whether or not they are present, such as "force majeure") have been correctly applied,
- whether the general principles of law have been respected, such as equality, legal certainty, proportionality...,
- of course, also whether the requirements and the "determining facts" have been correctly applied: Did the family really live near the school? Was it a large family or a single parent family?

However, once all this has been checked, the judge will not be able to assess why, out of all the options, A, B, C, D... has been chosen. This is so because, as we mentioned before, it is a matter of choosing between alternatives that are equally valid for the law and, therefore, are not subject to judicial review.

On the other hand, technical discretion is also reviewable. A judicial review can also be undertaken if a technical error committed is proven by evidence.

The decisions of a collegial body exercising technical discretion (a commission composed of technical experts, for example, an evaluation commission for the awarding of teaching positions or research projects) are presumed to be correct, unless a serious and manifest error (arbitrariness) is demonstrated.

Finally, judicial control of political discretion is limited, since it is a political decision and therefore subjective. However, it will not be exempt from legal control in terms of legal compliance, in particular, respect for human rights.

3.4 Automate (or not) the discretion to act

The automation of administrative decisions, even if they are regulated, is not an easy task because a huge part of relevant evidence is not directly deducible in the available facts (with some exceptions, e.g., when it comes to computing five years of active service to recognize a five-year teaching period in Spanish Universities). Most of the procedures require an interpretation of the legal rules in order to be applied, even if only regulated powers are exercised. In Spain there is a doctrinal majority (for the moment) that does not admit the use of fully automated decisions (without human intervention) when discretionary powers are exercised. Other legal systems do not admit it at all (Moreno Rebato, 2021; Perry, 2017; Huggins, 2021).

By consequence, there will be administrative procedures that can be 100% automated and others that can be partially automated and open a range of options (discretionary aspects). In this paper we propose a framework that automates the verification of the regulated issues but also offers (duly justified) the different legal options among which the competent body can apply discretion. In particular, with regard to the example of the allocation of school places discussed in the next Section, the discretionary part is attributed to the educational centers, which are administrative management bodies without political relevance.

4 s(LAW): A Legal Reasoner

In this section we present the other two contributions of the work:[3] (i) a set of patterns to translate legal rules into ASP, and to generate readable justifications in natural language, and (ii) a framework for modeling, reasoning, and justifying conclusions based on the evidence provided by the user and the applicable law, representing discretion, ambiguity and/or incomplete information (key concepts in legal cases).

4.1 Running example

Let us consider the following use case to explain the set of patterns and the framework we are proposing.

As we mentioned before, the first case to be modeled in this paper refers to the allocation of school places. Although there is freedom of choice of educational center, when the number of applications for a given center is greater than the number of places, the regulations establish ordering criteria to assign points, such as the priority and complementary criteria:

- Among the *priority criteria* are the fact of having siblings enrolled in the same center, the proximity of the family home or place of work of the parents or legal guardians of the student, and the income of the family unit (if they are beneficiaries of a minimum living income or a minimum insertion income).
- On the other hand, *complementary criteria* comprise facts like belonging to a large or single-parent family, parents working at the center, the existence of a disability of the student, being a victim of gender violence or terrorism, and parents who have been former students of the center.

Within these criteria, a small discretionary margin is attributed for each center to award points for “other circumstances” that may be established by each center, which may coincide with any of the above or which may be decided at the discretion of each center. A series of tie-breaking criteria are also specified. This information is not known because it is not known whether or not there are ties. Once this information is known, it is resolved by drawing lots. Finally, it is established that in “specific cases” that are necessary to meet educational needs, the number of groups and school units may be increased, e.g.,

[3]Preliminary results has been presented by Arias et al (2021)

a case of "force majeure" (a meteorological catastrophe that makes it impossible to open or use a certain center) and, all this, because always and in any case, the schooling of all students must be guaranteed (even if it is a different center than the one initially chosen).

It is, therefore, a fairly regulated procedure, in which only a small percentage of points are at the discretion of the school.

4.2 Patterns to translate law into ASP

The translation of legal rules into logic predicates has been considered a straightforward task for many years. However, the translation of ambiguity and/or discretion concepts required the help of an expert in law and/or in the field of application, in order to specify only one interpretation and/or decision.

Let us use the encoding of the procedure for the adjudication of school places in the CM (Fig. 1) to explain the following patterns:

Requirement For Applying

These are the most common constructions in legal articles, and we consider two main patterns:

- Disjunction of requirements, e.g., "s/he obtains a school place if one of the following common requirements are met". This is expressed by separating each requirement in different clauses, see Fig. 1 lines 9, 12, and 19:
- Conjunction of requirements, e.g., "In addition, some of the specific requirements must be met", which is translated into a single clause where the comma `','` means *and*, see Fig. 1 lines 5-7:

Exceptions For Applying

As we mentioned before, a legal article is, in general, a default rule subject to possible exceptions. In s(CASP) the exceptions can be encoded using negation as failure. For example, Fig. 1 lines 2-4 shows the translation of "It will be possible to obtain a school place if the requirement is met and there is no exception" and then, the compiler of s(CASP) would generate its dual, i.e., `not exception`, by collecting and checking that no exceptions hold:

```
not exception :- not exception_1, ..., not exception_n.
```

where `not exception_i` is a new predicate name that identified the dual of the ith exception. For the sake of brevity let us omit the explanation of how the compiler generates the dual for each exception (see (Marple et al, 2017; Arias et al, 2018) for details). Fig. 1 lines 46-57 shows the translation of the unique exception defined in our running example: "Students coming from non-bilingual public schools, who apply for a place in English language bilingual schools and who wish to study in the Bilingual Section, need to accredit a level of English in the four skills equivalent to level B1 for $1^{st}/2^{nd}$ ESO, and to level B2 for $3^{rd}/4^{th}$ ESO".

```
%% Obtain a school place if...
obtain_place(St) :-
  met_requirement(St),
  not exception(St).
met_requirement(St) :-
  met_common_requirement(St),
  met_specific_requirement(St).
%% Common requirements:
met_common_requirement(St) :-
  large_family(St).

met_common_requirement(St) :-
  recipient_social_benefits(St).
recipient_social_benefits(St) :-
  renta_minima_insercion(St).
recipient_social_benefits(St) :-
  ingreso_minimo_vital(St).

met_common_requirement(St) :-
  disability_status(St).
disability_status(St) :-
  disabled_parent(St).
disability_status(St) :-
  disabled_sibling(St).
%%  Specific requirements:
met_specific_requirement(St) :-
  sibling_enroll_center(St).
met_specific_requirement(St) :-
  legal_guardian_work_center(St).

met_specific_requirement(St) :-
  relative_former_student(St).

met_specific_requirement(St) :-
  school_proximity(St).
school_proximity(St) :-
  same_education_district(St).
school_proximity(St) :-
  not same_education_district(St),
  force_majeure.     % Ambiguity
force_majeure :-
  not n_force_majeure.
n_force_majeure :-
  not force_majeure.
%% Exceptions:
exception(St) :-
  come_non_bilingual(St),
  want_bilingual_section(St,Course),
  not accredit_english(St,Course).
accredit_english(St,'1st ESO') :-
  b1_certificate(St).
accredit_english(St,'2nd ESO') :-
  b1_certificate(St).
accredit_english(St,'3rd ESO') :-
  b2_certificate(St).
accredit_english(St,'4th ESO') :-
  b2_certificate(St).
%% Discretion To Act:
obtain_place(St) :-
  not met_requirement(St),
  met_complement_criterion(St,CC).
obtain_place(St) :-
  met_requirement(St), exception(St),
  met_complement_criterion(St,CC).

met_complement_criterion(St,CC) :-
  school_criteria(St,CC),
  purpose(CC), not unlawful(CC),
  not n_met_complement_criterion(St,CC).
n_met_complement_criterion(St,CC) :-
  not met_complement_criterion(St,CC).
purpose(CC) :-
  promote_diversity(CC).
unlawful(CC) :-
  sex_discrimination(CC).
unlawful(CC) :-
  race_discrimination(CC).
unlawful(CC) :-
  religion_discrimination(CC).

promote_diversity(foreign_student).
promote_diversity(specific_etnia).
race_discrimination(specific_etnia).

school_criteria(St,foreign_student) :-
  foreign_student(St).
school_criteria(St,specific_etnia) :-
  specific_etnia(St).
```

Fig. 1 Translation of the procedure for awarding school places under s(LAW).

Ambiguity

Ambiguity occurs when some aspects of the law can be interpreted in different ways. For example, "proximity to the family or work address" is a specific and defined requirement based on the distribution by educational districts. However, in case of *force majeure*, students from an education district may be reassigned to a school from another district. Fig. 1 lines 34-44 encode this scenario allowing evaluation without having to determine a priori the force majeure circumstances necessary to justify the reassignment of students. This pattern generates a model where `force_majeure` is assumed to hold and another model where there is *no* evidence that `force_majeure` holds.

Discretion To Act

Discretion to act introduces the possibility of choosing between different options that we intend to model by generating multiple models. Implementations based on Prolog compute a single, canonical model, and therefore, bypass this nondeterminism by selecting one interpretation. The discretion to act can be considered as a ground or an exception following the previous patterns. For example, Fig. 1 lines 59-79 shows the translation of the discretion to act rule: "The School Council may add another complementary criterion". The resulting encoding uses predicates in which the variable `CC` can be instantiated with different values. This feature allows us to reuse some of the clauses without repeating them, i.e., the clauses in lines 59-79 are generic, while clauses 81-88 specify the ground and exceptions of the criteria added by a particular school. Clauses in lines 66-71 generate two possible models if the discretion to act is exercised according to the purpose / intention of the law and it is not unlawful. In one model the complementary criterion is applied and in the other it does not. Then, clauses in lines 86-88 state the cases in which the discretion to act has a purpose and/or is unlawful.

Unknown Information

The use of default negation may introduce unexpected results in the absence of information (positive and/or negative). Therefore, in many cases the desirable behavior should capture the absence of information by generating different models depending on the relevant information. For example, it may be unclear whether the documents we have to certify that we are a `large_family` are valid or not, so we avoid introducing that information and the reasoner would reason assuming both scenarios. To state that some information is certain we would use the predicate `evidence/2`, e.g., `evidence(st01, large_family)` means that student 1 has the condition of large family. Additionally, s(LAW) would provide *strong* negation, denoted with `'-'`, to specify that we have evidences supporting the falsehood of some information, e.g., `-evidence(st04,large_family)` means that student 4 does not have the condition of large family.

```
1  #include('ArticleESO.pl').
2  #include('ArticleESO.pred.pl').
3
4  come_non_bilingual(St).
5  want_bilingual_section(St,'2nd ESO').
6
7  student(st01).
8  evidence(st01, large_family).
9  evidence(st01, renta_minima_insercion).
10 evidence(st01, sibling_enroll_center).
11 evidence(st01, same_education_district).
12 evidence(st01, b1_certificate).
13 -evidence(st01, foreign_student).
14 -evidence(st01, specific_etnia).
15
16 student(st02).
17 ...
```

Fig. 2 First lines of the file `students.pl`.

4.3 Description of s(LAW)

s(LAW), built on top of s(CASP), is composed by three modules: the first contains the *articles*, the second contains *explanations* to generate readable justifications, and the third one contains *evidence* from a set of students. In our running example:

ArticleESO.pl

Contains the legislation rules in Fig. 1 following the patterns described in Section 4.2.

ArticleESO.pre.pl

Contains the natural language patterns for the predicates that are relevant to provide readable justifications of the conclusions inferred by s(LAW). The directive `#pred` defines the natural language patterns, e.g.:

```
#pred obtain_place(St) :: '@(St) may obtain a school place'.
```

Note that the natural language pattern is customized based on the id of the student, `@(St)`.

Additionally, to facilitate the understanding of the code we can obtain a readable code (in natural language) by invoking `scasp --code --human`.

Students.pl

Fig. 2 shows the encoding of the module `students.pl` corresponding to a set of 6 student. This last module encodes the evidence of 6 students and links them with the previous modules `ArticleESO.pl` and `ArticleESO.pred.pl` (lines 1-2). The predicates `evidence/2` and `-evidence/2` (explained in Section 4.2) are used to specify the known information, second argument, (positive or negative evidences) for each student, first argument. For the sake of brevity, let us handle as *unknown* evidence corresponding to: `large_family`, `renta_minima_insercion`, `sibling_enroll_center`, `same_education_district`, `b1_certificate`, `foreign_student`, and `specific_etnia`. Fig. 2 lines 7-14 provide the known information corresponding to the first student. Note that we consider that all students, coming from

Table 1 Case of different students evaluated using s(LAW).
Note: '+' is a positive evidence, '−' is a negative evidence, '?' means unkown.

	st01	st02	st03	st04	st05	st06
`large_family`	+	+	+	−	−	−
`renta_minima_insercion`	+	+	+	?	−	−
`sibling_enroll_center`	+	+	−	+	−	−
`same_education_district`	+	+	−	+	−	−
`b1_certificate`	+	−	+	?	−	−
`foreign_student`	−	−	−	−	+	−
`specific_etnia`	−	−	−	−	−	+
`?- obtain_place(stXX)`	**yes**	no	**yes**	**yes**	**yes**	no

non-bilingual public schools, apply for a place in English language bilingual schools and wish to study in the Bilingual Section (Fig. 2 lines 4-5).

5 Explainable Reasoning

The modules of s(LAW) are implemented under s(CASP) version 0.22.12.14 (https://gitlab.software.imdea.org/ciao-lang/sCASP), that runs under Ciao Prolog version 1.19-480. (http://ciao-lang.org/). The benchmarks used in this section are available at http://platon.etsii.urjc.es/~jarias/papers/slaw-ailaw23 and were run on a MacOS 13.2.1 laptop with an Apple Core M2.

A priori Deduction: Consider we run our reasoner s(LAW) in the interactive mode to reason about the six students by invoking:

```
scasp -i --tree --human --short --pos students.pl
```

Then, we launch the queries to obtain conclusions from the reasoner. Table 1 shows the data corresponding to the candidates and the conclusion generated by s(LAW) for the query `?- obtain_place(St)`. Students 1, 3, 4, and 5 may obtain a place at the school[4] while students 2 and 6 do not:

- Student 1: Fig. 2 contains the information corresponding to this student. Since s/he meets common and specific requirements and avoids the exception (having level b1 in English), the evaluation returns the partial model:

  ```
  { obtain_place(st01), large_family(st01), sibling_enroll_center(st01),
    come_non_bilingual(st01), want_bilingual_section(st01,2nd ESO),
    b1_certificate(st01) }
  ```

  and the corresponding justification shown in Fig. 3.

[4]We discuss later on that, under different assumptions, students 3 and 4 do not obtain a place.

```
st01 may obtain a school place, because
    a common requirement is met, because
        st01 is part of a large family.
    a specific requirement is met, because
        st01 has siblings enrolled in the center.
    there is no evidence that an exception applies, because
        st01 came from a non-bilingual public school, and
        st01 came from a non-bilingual public school, justified above, and
        st01 wish to study 2nd ESO in the Bilingual Section, and
        st01 accredit required level of English for 2nd ESO, because
            in the four skills certificate level b1.
```

Fig. 3 Justification in Natural Language for the query `?- obtain_place(st01)`.

```
st03 may obtain a school place, because
    a common requirement is met, because
        st03 is part of a large family.
    a specific requirement is met, because
        the school is near the family or work, because
            'force_majeure' holds, because
                it is assumed that 'force_majeure' holds.
    there is no evidence that an exception applies, because
        st03 came from a non-bilingual public school, and
        st03 came from a non-bilingual public school, justified above, and
        st03 wish to study 2nd ESO in the Bilingual Section, and
        st03 accredit required level of English for 2nd ESO, because
            in the four skills certificate level b1.
```

Fig. 4 Justification in Natural Language for the query `?- obtain_place(st03)`.

- Student 2: meets common and specific requirements but has to be rejected because s/he does not accredit level b1 in English (in Table 1 that is identified with a '−' in the corresponding column/row). Therefore, for the query `obtain_place(st02)` s(LAW) returns **no model**.
- Student 3: meets common requirements and avoids the exception. But s/he does not meet any specific requirement (`sibling_enroll_center` or `same_education_district`). Nevertheless, by assuming `force_majeure` s/he also meets a specific requirement `school_proximity`, so s(CASP) returns the partial model:

  ```
  { obtain_place(st03),  large_family(st03),  school_proximity(st03),
    force_majeure,  come_non_bilingual(st03),
    want_bilingual_section(st03,2nd ESO),  b1_certificate(st03) }
  ```

  and we see in the corresponding justification (see Fig. 4) that `school_proximity` holds by assuming `force_majeure`.
- Student 4: in this use-case there is absence of information regarding the "renta minima de insercion" (`renta_minima_insercion`) and the English certificate (`b1_certificate`), which are marked with ? in Table 1. The

```
st05 may obtain a school place, because
    the criterion foreign_student is met, because
        st05 meets the criteria foreign_student, because
            st05 is a foreign student.
        foreign_student follows the purpose of the procedure, because
            foreign_student promotes the diversity.
        it is assumed that the criterion foreign_student is met.
```

Fig. 5 Justification in Natural Language for the query `?- obtain_place(st05)`.

partial model returned assumes that the truth values for these pieces of information are true:

```
{ obtain_place(st04), renta_minima_insercion(st04),
  sibling_enroll_center(st04), not exception(st04), come_non_bilingual(st04),
  want_bilingual_section(st04,2nd ESO), b1_certificate(st04) }
```

therefore, based on that assumption the student would obtain a place.

- Student 5: now let's consider that there is a school with a complementary criterion for foreign students and therefore, since the student is a foreigner, s/he obtains a place. Fig. 5 shows the justification for this example.
- Student 6: now we consider that the complementary criterion is for student of a specific etnia, and that student 6 belongs to this ethnic group. However, this criterion, `specific_etnia`, cannot be applied because it discriminates by race and, thus, is unlawful. Therefore, **no model** is returned, and the student does not obtain a place.

A posteriori Deduction: The main advantage of s(LAW) is its ability to generate justifications not only for positive but also for negative information. To extract a justification including the negated literals we include the flag `--neg` in the invocation:

```
scasp -i --tree --human --short --neg students.pl
```

This ability would allow us to analyze the reason for a specific inference and/or to determine which are the requirements needed to obtain a specific conclusion:

- For student 2, the query `?- not obtain_place` returns a partial model and the justification (see Fig. 6) supporting that the student does not obtain a place. While student 2 met a common and a specific requirement, s/he does not accredit the required level of English and s/he does not meet any complementary criterion. Note that the last check is done for every possible complementary criterion, i.e., by checking that there is no evidence for `Var0` not equal `foreign_student`, nor `specific_etnia`, and by checking that there is no evidence for both of them.
- For student 3, the query `?- not force_majeure, obtain_place(st03)` avoids the assumption of force majeure and therefore, s(LAW) returns **no model**. Note that student 3 does not meet any specific requirement so while

```
there is no evidence that st02 may obtain a school place, because
    a common requirement is met, because
        st02 is part of a large family.
    a specific requirement is met, because
        st02 has siblings enrolled in the center.
    an exception applies, because
        st02 came from a non-bilingual public school, and
        st02 wish to study 2nd ESO in the Bilingual Section, and
        there is no evidence that st02 accredit required level of English for 2nd ESO, because
            there is no evidence that in the four skills certificate level b1.
    an exception applies, justified above, and
    there is no evidence that the complementary criterion Var0 not equal foreign_student, nor
      ↪ specific_etnia is met, because
        there is no evidence that st02 meets the criteria Var0 not equal foreign_[...]
    there is no evidence that the complementary criterion foreign_student is met, because
        there is no evidence that st02 meets the criteria foreign_student, because
            there is no evidence that st02 is a foreign student.
    there is no evidence that the complementary criterion specific_etnia is met, because
        there is no evidence that st02 meets the criteria specific_etnia, because
            there is no evidence that st02 belongs to a specific etnia.
```

Fig. 6 Justification in Natural Language for the query `?- not obtain_place(st02)`.

```
there is no evidence that st04 may obtain a school place, because
    there is no evidence that a common requirement is met, because
        there is no evidence that st04 is part of a large family, and
        there is no evidence that st04 is a recipient of the RMI, because
            it is assumed that there is no evidence that st04 is a recipient of the RMI.
        there is no evidence that a parent or sibling of st04 has disability status.
    there is no evidence that the complementary criterion Var0 not equal foreign_student, nor
      ↪ specific_etnia is met, because
        there is no evidence that st04 meets the criteria Var0 not equal foreign_[...]
    there is no evidence that the complementary criterion foreign_student is met, because
        there is no evidence that st04 meets the criteria foreign_student, because
            there is no evidence that st04 is a foreign student.
    there is no evidence that the complementary criterion specific_etnia is met, because
        there is no evidence that st04 meets the criteria specific_etnia, because
            there is no evidence that st04 belongs to a specific etnia.
```

Fig. 7 Justification in Natural Language for the query `?- not obtain_place(st04)`.

under the assumption of `force_majeure` s/he meets the specific requirement for `school_proximity`, without that assumption s/he does not and therefore, s/he does not obtain a place.

- For student 4, the query `?- not obtain_place(st04)` succeeds considering the assumptions for which this student does not obtain a place. For example, Fig. 7 shows the justification where it is assumed that `renta_minima_insercion` does not holds.

```
1  there is no evidence that st06 may obtain a school place, because
2      there is no evidence that a common requirement is met, because
3          there is no evidence that st06 is part of a large family, and
4          there is no evidence that st06 is a recipient of the RMI, and
5          there is no evidence that a parent or sibling of st06 has disability status.
6      there is no evidence that the complementary criterion Var0 not equal foreign_student, nor
           ↪ specific_etnia is met, because
7          there is no evidence that st06 meets the criteria Var0 not equal foreign_[...]
8      there is no evidence that the complementary criterion foreign_student is met, because
9          there is no evidence that st06 meets the criteria foreign_student, because
10             there is no evidence that st06 is a foreign student.
11     there is no evidence that the complementary criterion specific_etnia is met, because
12         st06 meets the criteria specific_etnia, because
13             st06 belongs to a specific etnia.
14         specific_etnia follows the purpose of the procedure, because
15             specific_etnia promotes the diversity.
16         specific_etnia is illegal, because
17             specific_etnia discriminates based on race.
```

Fig. 8 Justification in Natural Language for the query `?- not obtain_place(st06)`.

- For student 6, Fig. 8 shows the justification of the query `?- not obtain_place(st06)` so we can analyze more in detail why this student is rejected. While the complementary criteria for student 5 (`foreign_student`) is similar to `specific_etnia`, the justification tree shows that student 6 does not obtain a place because the complementary criterion `specific_etnia` is illegal due to `race_discrimination`, see Fig. 8 lines 16-17.

Additionally, we can collect the partial models, in which the school place is or is not obtained, together with their justification and analyze "Epistemic Specifications" (Gelfond, 1994), that is, what is true in all/some models, which partial models share certain assumptions, etc. This reasoning makes it possible to detect the missing information that would change the decision from "not obtained" (or "obtained" under some assumptions) to "obtained". Note that, by introducing the new evidence, the resulting justification of s(LAW) provides an explanation in which these evidence are used to support the decision.

6 Related Work

In attempting to model how legal discretion is exercised, Schild and Zeleznikow (2005) and Kannai et al (2007) discuss the notion of open texture, the notion of delimitation of legal domains, and whether decisions made in a given domain are binary. They propose to model discretionary decision-making using three independent axes: bounded (B) and unbounded (U), defined (D) and undefined (U), and binary (B) and continuous (C). The resulting classification makes it possible to select the adequate inferencing techniques for each octant:

1. **Bounded, Defined, Binary (BDB)** All issues and rules are known, and the decision is binary. A rule-based approach can be use, e.g., to model the domain of driving offences where drivers can lose their licence by being drunk based on the blood alcohol level.
2. **Bounded, Defined, Continuous (BDC)** All issues and rules are known, but the decision is continuous. In these cases, it is necessary to define systems, such as the US Sentencing Guidelines, to alleviate sentencing disparities.
3. **Bounded, Undefined, Binary (BUB)** The issues are known and the decision is binary, but no knowledge is available about how they should be combined. Knowledge discovery, based on previous landmark cases, can be automated, However, machine learning techniques require such a large number of cases that it is rarely available in any legal domain.
4. **Bounded, Undefined, Continuous (BUC)** The task of distributing property following divorce is, in general, an example in which the relative importance of the different factors is not specified, and crucial terms are not defined. In this scenario of knowledge discovery, to alleviate disparities, continuous values are involved.
5. **Unbounded, Defined, Binary (UDB)** Only some issues and the way they combine are known and the decision is binary, e.g., determine if a driver is guilty under the drink-driving regulations based on the notion of dangerous driving.
6. **Unbounded, Defined, Continuous (UDC)** Determine the sentence of a guilty driver under the drink-driving regulations based on the notion of dangerous driving.
7. **Unbounded, Undefined, Binary (UUB)** Legal decision-makers in these domains exercise a great degree of discretion with a binary output, therefore, each decision have a great impact, e.g., a refugee review decision has impact on both the applicant and relevant family.
8. **Unbounded, Undefined, Continuous (UUC)** Deciding on the residence of the children of the marriage after divorce, in a legislation that tries to move away from the "all or nothing", mitigates the risk of unfairness but requires a greater effort of consistency with previous judgments.

Tasks in the octants 1 and 2 can be modeled using proposals based on logic programming languages such as those by Ramakrishna et al (2016) and Sergot et al (1986) using deterministic rules.

To model unbounded and/or undefined tasks, in octants 3-8, non-monotonic reasoning is needed, so we propose the use of Answer Set Programming (ASP), a successful paradigm for developing intelligent applications and has attracted much attention due to its expressiveness, ability to represent knowledge, incorporate non-monotonicity, and model combinatorial problems. For example, Aravanis et al (2018) propose the integration of AI and Law by providing a knowledge-based system, capable of representing and non-monotonically reasoning about legal knowledge, thanks to the use of the negation as failure of ASP. Additionally, Purnell and Schwitter (2022) present a methodology to write and test smart contracts using the online editor SDEv4[5] that uses ASP to represent both the ontology and the legal logic of contracts.

However, most ASP systems follow bottom-up executions that require a grounding phase where the variables of the program are replaced with their possible values. During the grounding phase, links between variables are lost and therefore an explanation framework for these systems must face many challenges to provide a concise justification of why a specific answer set satisfies the rules (and which rules). The most relevant approaches are: off-line and on-line justifications (Pontelli et al, 2009); Causal Graph Justification (Cabalar et al, 2014); and Labeled ABA-Based Answer Set Justification (LABAS) (Schulz and Toni, 2016). However, these approaches are applied to grounded versions of the programs, i.e., non-ground programs have to be grounded, and they may produce unwieldy justifications when the non-ground program has uninterpreted functions, consults large databases and/or requires the representation of dense domains (Arias et al, 2018).

By contrast, systems that follow a top-down execution can trace which rules have been used to obtain the answers more easily. One such system is ErgoAI (https://coherentknowledge.com), based on XSB (Swift and Warren, 2012), that generates justification trees for programs with variables. ErgoAI has been applied to analyze streams of financial regulatory and policy compliance in near real-time providing explanations in English that are fully detailed and interactively navigable. However, default negation in ErgoAI is based on the well-founded semantics (Gelder et al, 1991) and therefore ErgoAI is not a framework that allows the representation of ambiguity and/or discretion.

Moreover, in the context of Rules as Code, there are proposals that exploit the advantages of s(CASP). Rule as Code is an interdisciplinary, technology-enabled approach to the development of rules that is designed to improve policy outcomes. According to Wong (2020), Rules as Code features multidisciplinary teams of lawyers, policy experts, computer programmers, and others who draft rules in natural languages and code at the same time. It is rooted in the effort of automating legal reasoning put forward by Sergot et al (1986), who formalised the British Nationality Act using Prolog, in order to determine whether or not a person is eligible for British nationality based on various criteria. Some relevant proposals that are exploiting s(CASP) in the context of the Rules as Code paradigm are the following:

- L4 is a domain-specific language developed in the context of the CCLAW project at the Singapore Management University by Lim et al (2022). It is executed using SMT solvers, but more recently it can also be translated into s(CASP). Note that L4 focuses on the formalization of the law avoiding the ambiguities of natural languages, and its use of s(CASP) is restricted to the possibility of reasoning with incomplete information.
- Blawx is a user-friendly Rules as Code interface on the web developed by Morris (2021). Initially, its back-end was ErgoAI, but recently it has also

[5] SDEv4 is available at http://130.56.246.229

been re-implemented using s(CASP). As a consequence, Blawx provides justifications in natural languages for positive and negative queries.

In addition, there are proposals focusing on the automation of the translation of statutes from natural language into code. The proposal by Zhou et al (2022) integrates Natural Language Processing and context-free grammar to parse labeled sentences into a language-independent tree structure (from which computable checking rules can be generated). The program *ylegis*, by Mowbray et al (2023), converts the text of legislation into *script rules* that reflect the legislation's structure, i.e., the drafting of legislation could be changed to make it directly readable and understandable by humans and also usable by machines.

The work presented in this paper is not concerned with the automated translation of legal text, but it can explicitly represent and reason with the vague concepts that it contains. In particular, it can deal with open-texture terms that appear due to the use of natural language and give rise to competing interpretations. So, when encoding natural language legal rules, choices regarding the correct interpretation of such terms need not be taken at design-time.

CATALA by Huttner and Merigoux (2022) is perhaps the most relevant methodology in practice to build, validate, and explain legal text. It has been used to implement different statutes, such as the French family benefits[6] and, thanks to its compiler, the resulting implementation can be translated into general-purpose programming languages, such as Python, Java, among others (see the paper by Merigoux et al (2021) for details). However, it can neither explicitly represent vague concepts and incomplete information, nor can it answer negated queries. Therefore, the addition of the translation patterns proposed in this work (especially with regard to discretionality) may improve the expressiveness of CATALA and other practical systems.

Finally, it seems worthwhile relating the present proposal to recent works in the fields of case-based reasoning (CBR) and computational argumentation that address similar issues to the ones tackled by this paper. Rissland et al (2005) motivate that Law is a particularly interesting domain for CBR researchers given that it is largely about cases, and Anglo-American law is precedent-based (its judicial standard mandates that similar cases should be decided similarly). They present several solutions for extracting relevant information from legal cases, and for reasoning based on cases and/or rules. However, none of the surveyed systems explicitly support presenting and reasoning about vague concepts in the context of administrative law. The survey put forward by Bench-Capon (2012) focuses on the question *what makes an argument persuasive?*, and describes the progress made in AI and Law to this respect over the last 25 years. Answering this question is not trivial because, as we have argued before, often legislators use vague terms in order to make their general intentions clear, but leaving it to the courts to establish the concrete limits of the concept in light of the facts of each case. Abstract argumentation

[6] Example available at https://catala-lang.org/en/examples/family-benefits

presents arguments in a framework in which they can be entirely abstract, related only by a binary relation of attack. Thus, it can be used to evaluate the admissibility of different interpretations of vague terms. Therefore, this approach can be conceived as orthogonal to the proposal put forward in this paper. The work by Licato (2021) has a similar aim. It faces the existence of rules with open-textured terms which imbues them with exponential number of interpretations, and tries to answer the question *given an open-textured rule, what is its correct interpretation?*. Billi et al (2021) compare different logic-based proposals that can be used to perform defeasible reasoning and provide a tool, called Arg2P, which implements different approaches and semantics in a single environment, supporting preferences over rules.

7 Conclusions

The present work is an example for the fruitful collaboration of computer scientists and lawyers in the context of legal knowledge engineering (Susskind, 2017). In principle, there are two major ways of translating, modeling and applying legal rules by converting them into computer language. On the one hand, it may be the law that is adjusted to computer languages, in which case the role of the jurist would be to reformulate the legal rules, trying to reduce indeterminate legal concepts, discretionality and other ambiguous elements, approaching what is called computational law (Sergot et al, 1986; Genesereth, 2015; Ramakrishna et al, 2016; Liebwald, 2015; Branting, 2017). An example is the translation of Rule 34 of the Legal Profession (Professional Conduct) Rules 2015 of Singapore, which proceeds to amend the word "business" in the statutory text.[7] On the other hand, the computer languages may be capable of modeling the legal language including ambiguities, its indeterminate legal concepts and discretionality. With the present work we intended to advance in this second possibility through discretionary decision support systems.[8]

In particular, in this paper we have shown that using goal-directed answer set programming, s(LAW) is capable of modeling vague concepts such as discretion and ambiguity: The adjudication of school places is a task in the third octant (BUB). It has a bounded domain with a certain degree of vagueness (it is not 100% defined) and is a binary decision making. The deduction based on s(LAW) allows: the consideration of different conclusions (multiple models) which can be analyzed by humans thanks to the justification generated in natural language; and the reasoning about the set of these conclusions/models. To the best of our knowledge, s(LAW) is the only system that exhibits the property of modelling vague concepts.[9] Note that none of the proposals mentioned

[7] https://github.com/smucclaw/r34_sCASP.

[8] In the near future, it may become necessary advance in the direction of the first option as well. In this sense, we note the need to offer mixed curricula, Law and Artificial Intelligence (Rodríguez-García and Moreno-Rebato, 2018).

[9] On January 14^{th}, 2021, Dr. Robert Kowalski explained how they bypassed in (Sergot et al, 1986) the representation of vague concepts such as *without undue delay* (Kowalski, 2021, 1:20:15, 1:26:00).

in Section 6 is able to reason (and justify) considering different assumptions regarding such concepts.

A limitation of our proposal concerns its scalability. The awarding of school places in centers supported with public funds in the "Comunidad de Madrid" presented in this paper is a relatively small regulation. So, the number of rules that need to be drafted (design complexity) and reasoned with (computational complexity) is limited. Still, the codification of laws on a large scale (e.g., the British Nationality Act, as presented by Sergot et al (1986)) constitutes an engineering problem whose solution requires a large number of experts in law and logic to formalize and verify large sets of rules. Also, problems related to the run-time complexity of automated inference on such large rule sets can become an issue. Still, we believe that there are sufficient small and medium scale regulations in the context of administrative procedures that could benefit from our approach. In fact, we are currently trying to identify appropriate domains with the help of start-ups specialised in the field.

Additional future work unfolds among two major lines. First, we aim at augmenting the expressive power of our language for modeling the legislation by allowing for non-binary data. For example, we will add the option of assigning different weights to the evaluation criteria used in the procedure for the adjudication of school places in centers supported with public funds. We will also support continuous domains for some features, such as school proximity. This information can then be exploited by the constraint solver underlying s(CASP). Second, we want to explore the combination of different rule-based techniques to create a *hybrid* model where (i) the legislation is modeled as rules, (ii) previous sentences are stored in databases (non-monotonic reasoning is needed to ensure consistency), and (iii) inductive logic programming is applied (instead of "traditional" machine learning) to extract rules from a reduced number of landmark cases.

References

Aravanis TI, Demiris K, Peppas P (2018) Legal Reasoning in Answer Set Programming. In: IEEE 30th International Conference on Tools with Artificial Intelligence, ICTAI 2018. IEEE, pp 302–306, https://doi.org/10.1109/ICTAI.2018.00055

Arias J, Carro M, Salazar E, et al (2018) Constraint Answer Set Programming without Grounding. Theory and Practice of Logic Programming 18(3-4):337–354. https://doi.org/10.1017/S1471068418000285

Arias J, Carro M, Chen Z, et al (2020) Justifications for goal-directed constraint answer set programming. In: Proceedings 36th International Conference on Logic Programming (Technical Communications), EPTCS, vol 325. Open Publishing Association, pp 59–72, https://doi.org/10.4204/EPTCS.325.12

Arias J, Moreno-Rebato M, Rodriguez-García JA, et al (2021) Modeling Administrative Discretion Using Goal-Directed Answer Set Programming. In: Advances in Artificial Intelligence, CAEPIA 20/21. Springer International Publishing, Cham, pp 258–267, https://doi.org/10.1007/978-3-030-85713-4_25

Bench-Capon T (2012) Open Texture and Argumentation: what makes an argument persuasive? In: Logic Programs, Norms and Action: Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday. Springer, p 220–233, https://doi.org/10.1007/978-3-642-29414-3_12

Billi M, Calegari R, Contissa G, et al (2021) Argumentation and defeasible reasoning in the law. J 4(4):897–914. https://doi.org/10.3390/j4040061

Branting L (2017) Data-centric and logic-based models for automated legal problem solving. Artificial Intelligence and Law 25(1):5–27. https://doi.org/10.1007/s10506-017-9193-x

Cabalar P, Fandinno J, Fink M (2014) Causal Graph Justifications of Logic Programs. Theory and Practice of Logic Programming 14(4-5):603–618. https://doi.org/10.1017/S1471068414000234

Cobbe J (2019) Administrative law and the machines of government: judicial review of automated public-sector decision-making. Legal Studies 39(4):636–655. https://doi.org/10.1017/lst.2019.9

Cosculluela Montaner L (2021) Manual de Derecho Administrativo. Thomson Reuters Civitas pp 333–ff. ISBN: 978-84-1125-550-9

Garcia de Enterría E (1962) La lucha contra las inmunidades del poder en el derecho administrativo". Revista de Administración Pública 38:159–208. URL http://www.cepc.gob.es/sites/default/files/2021-12/222271962038159.pdf

Garcia de Enterría E, Fernández TR (2020) Curso de Derecho Administrativo. Civitas ISBN: 978-84-1125-553-0

Gelder AV, Ross K, Schlipf J (1991) The Well-Founded Semantics for General Logic Programs. Journal of the ACM 38:620–650. https://doi.org/10.1145/116825.116838

Gelfond M (1994) Logic programming and reasoning with incomplete information. Annals of mathematics and artificial intelligence 12(1):89–116. https://doi.org/10.1007/BF01530762

Gelfond M, Lifschitz V (1988) The Stable Model Semantics for Logic Programming. In: 5th International Conference on Logic Programming, pp 1070–1080, URL http://www.cse.unsw.edu.au/~cs4415/2010/resources/stable.pdf

Genesereth M (2015) Computational Law. The Cop in the Backseat. Codex: The Center for Legal Informatics, Stanford University

Huber H (1953) Niedergang des Rechts und Krise des Rechtsstaat. Festgabe für Z Giacometti, Zürich p 66

Huergo Lora A (2020) Una aproximación a los algoritmos desde el Derecho Administrativo, La regulación de los algoritmos. Aranzadi Thomson Reuters, ISBN: 978-84-1345-094-0

Huggins A (2021) Addressing disconnection: automated decision-making, administrative law and regulatory reform. UNSW Law Journal 44(3):1048–1077. https://doi.org/10.53637/WCGG2401

Huttner L, Merigoux D (2022) Catala: moving towards the future of legal expert systems. Artificial Intelligence and Law pp 1–24. https://doi.org/10.1007/s10506-022-09328-5

Kannai R, Schild U, Zeleznikow J (2007) Modeling the evolution of legal discretion. an artificial intelligence approach. Ratio Juris 20(4):530–558

Kowalski RA (2021) Logical English = Logic + English + Compupting. https://utdallas.app.box.com/s/ngsyloscj5sk24uh3axexxz451o74z0u, Hack-Reason Opening Ceremony. Last accessed 19 April 2021

Licato J (2021) How Should AI Interpret Rules? A Defense of Minimally Defeasible Interpretive Argumentation. arXiv URL https://arxiv.org/pdf/2110.13341.pdf

Liebwald D (2015) On transparent law, good legislation and accessibility to legal information: Towards an integrated legal information system. Artificial Intelligence and Law 23(3):301–314. https://doi.org/10.1007/s10506-015-9172-z

Lim HK, Mahajan A, Strecker M, et al (2022) Automating Defeasible Reasoning in Law with Answer Set Programming. In: Proceedings of the International Conference on Logic Programming 2022 Workshops (ICLP 2022), vol 3193. CEUR-WS.org, URL https://ceur-ws.org/Vol-3193/paper3GDE.pdf

Marple K, Salazar E, Gupta G (2017) Computing Stable Models of Normal Logic Programs Without Grounding. arXiv 1709.00501. URL http://arxiv.org/abs/1709.00501, https://arxiv.org/abs/1709.00501

Cerrillo i Martínez A (2020) ¿Son fiables las decisiones de las Administraciones públicas adoptadas por algoritmos? European review of digital administration & law 1(1):18–36

Merigoux D, Chataing N, Protzenko J (2021) Catala: a programming language for the law. Proceedings of the ACM on Programming Languages 5(ICFP):1–29. https://doi.org/10.1145/3473582

Moreno Rebato M (2021) Inteligencia artificial (Umbrales éticos, Derecho y Administraciones públicas), Thomson Reuters, Aranzadi, pp 120–ff. ISBN: 978-84-1391-029-1

Morris J (2021) Constraint answer set programming as a tool to improve legislative drafting: a rules as code experiment. In: ICAIL '21: Eighteenth International Conference for Artificial Intelligence and Law. ACM, pp 262–263, https://doi.org/10.1145/3462757.3466084

Mowbray A, Chung P, Greenleaf G (2023) Representing legislative Rules as Code: Reducing the problems of 'scaling up'. Computer Law & Security Review 48:105,772. https://doi.org/10.1016/j.clsr.2022.105772

Otenyo E (2016) Administrative Discretion. In: Farazmand A (ed) Global Encyclopedia of Public Administration, Public Policy, and Governance. Springer, https://doi.org/10.1007/978-3-319-31816-5_955-1

Perry M (2017) iDecide: Administrative Decision-Making in the Digital World. Australian Law Journal 91(1):29

Pontelli E, Son TC, El-Khatib O (2009) Justifications for Logic Programs under Answer Set Semantics. Theory and Practice of Logic Programming 9(1):1–56. https://doi.org/10.1017/S1471068408003633

Purnell K, Schwitter R (2022) User-Defined Smart Contracts Using Answer Set Programming. In: AI 2021: Advances in Artificial Intelligence - 34th Australasian Joint Conference, AI 2021, vol 13151. Springer, pp 291–303, https://doi.org/10.1007/978-3-030-97546-3_24

Ramakrishna S, Górski Ł, Paschke A (2016) A dialogue between a lawyer and computer scientist: the evaluation of knowledge transformation from legal text to computer-readable format. Applied Artificial Intelligence 30(3):216–232. https://doi.org/10.1080/08839514.2016.1156952

Rissland EL, Ashley KD, Branting LK (2005) Case-based reasoning and law. The Knowledge Engineering Review 20(3):293–298. https://doi.org/10.1017/S0269888906000701

Rodríguez-García JA, Moreno-Rebato M (2018) ¡El futuro ya está aquí! Derecho e inteligencia artificial. Revista Aranzadi de Derecho y nuevas tecnologías 48. ISSN: 1696-0351

Schild UJ, Zeleznikow J (2005) A taxonomy for modelling discretionary decision making in the legal domain. In: Proceedings of the 10th international conference on Artificial intelligence and law, pp 60–64, https://doi.org/10.1145/1165485.1165495

Schulz C, Toni F (2016) Justifying Answer Sets Using Argumentation. Theory and Practice of Logic Programming 16(1):59–110. https://doi.org/10.1017/S1471068414000702

Sergot MJ, Sadri F, Kowalski RA, et al (1986) The british nationality act as a logic program. Communications of the ACM 29(5):370–386. https://doi.org/10.1145/5689.5920

Sourdin T (2021) Judges, Technology and Artificial Intelligence: The Artificial Judge, Edward Elgar Publishing Limited, pp 228–ff. https://doi.org/10.4337/9781788978262

Suksi M (2021) Administrative due process when using automated decision-making in public administration: some notes from a finnish perspective. Artificial Intelligence and Law 29(1):87–110. https://doi.org/10.1007/s10506-020-09269-x

Susskind R (2017) Tomorrow's Lawyers: An Introduction to Your Future, 2nd edn. Oxford University Press, ISBN: 978-01-9879-663-3

Swift T, Warren DS (2012) XSB: Extending Prolog with Tabled Logic Programming. Theory and Practice of Logic Programming 12(1-2):157–187. https://doi.org/10.1017/S1471068411000500

Sánchez Morón M (2021) Derecho Administrativo. Parte General. Tecnos pp 91–ff. ISBN: 978-84-309-8529-6

Vestri G (2021) La inteligencia artificial ante el desafío de la transparencia algorítmica: Una aproximación desde la perspectiva jurídico-administrativa. Revista Aragonesa de Administración Pública (56):368–398

Wong MW (2020) Rules as code: Seven levels of digitisation. In: Research Collection Yong Pung How School Of Law. CCLAW-SMU, URL https://ink.library.smu.edu.sg/sol_research/3093/

Zhou YC, Zheng Z, Lin JR, et al (2022) Integrating NLP and context-free grammar for complex rule interpretation towards automated compliance checking. Computers in Industry 142:103,746. https://doi.org/10.1016/j.

compind.2022.103746