The LegAi Editor: A Tool for the Construction of Legal Knowledge Bases

Tomer Libal
University of Luxembourg
ORCiD ID: Tomer Libal https://orcid.org/0000-0003-3261-0180

Abstract. A key challenge in legal knowledge representation is the construction of formal knowledge bases. Such knowledge bases then allow for various applications such as searching and reasoning. In this paper we describe a web application for the annotation of legal texts. The result of the annotation is a formal representation of the legal texts in a form which might be used for searching and reasoning. The novelty of the tool is threefold: it provides a wizard which guides the user in the annotation process; it uses a high-level annotation language which is close to the language of the original text; and, it allows validation of the formal structure by allowing a simple comparison of the original and the annotated versions of the legal text.

Keywords. Legal Knowledge Representation, Automated Reasoning, Annotation Editor

1. Introduction

Information systems are playing an important role in helping people in a wide range of tasks, ranging from searching to decision-making.

One area in which such tools can contribute is the legal domain: New court cases and legislations are accumulated every day. In addition, international organizations like the European Union are constantly aiming at combining and integrating separate legal systems [4].

Approaches for searching over legal texts have long seen commercial success [17]. At the same time, some tools for reasoning over sets of norms have been developed, such as for business (e.g. [6]) and law (e.g. [12] and [9]).

Unlike the tools and methods for searching, which have seen a wide commercial success and plans are in place for supporting them, those for reasoning have seen a much smaller success, especially by legal practitioners.

One of the reasons for the relatively limited success of applications of automated reasoning in the legal domain is the lack of tools for the construction of legal knowledge bases, which can be used by non-logicians.

Various surveys of the available tools (e.g. [11]), have identified usability by domain experts as one of the main requirements. Nevertheless, most such tools fail in achieving this goal [15].

The most popular approach is by the use of programming languages (a classical example can be found in [14]). This approach has enjoyed success [3] in the last 40 years and is still popular today. Nevertheless, basic knowledge of logic programming is still required. In order to have a wider use of legal reasoning, other professionals, such as lawyers and jurists, should be able to use the tools.

Various attempts have been done in order to make such tools more accessible to legal practitioners (e.g. [7]). The advantage of these approaches is the ability to compile and test the correctness and quality of the knowledge.

Another approach, demonstrated in [13], is by the use of general formats and languages. Being of general formats, one can more easily utilize a language which is most suited for capturing the semantics of legal texts.

A disadvantage of all the approaches above is the lack of tools and methodologies for asserting the correctness of the logical representations of the legal texts. A discussion about the need can be found in [8][1].

Among existing validation results, one can find a methodology for building legal ontologies [10] and more concretely to our approach, one for validating formal representations of legal texts [1].

In this paper, we demonstrate the LegAi annotation editor. The focus of this editor is on the ability to validate the quality of the formalized legal texts, as well as on usability. Nevertheless, the editor uses a specific formal language, which can be transformed into first-order Deontic logics.

The paper will those be divided into three parts: we first describe the formal language used; in the second part, the annotation editor and its user interface are presented; we end by presenting an approach for ensuring the quality of the legislation.

2. The Legal Linguistic Templates

In the automated reasoning community there is a discussion whether the knowledge representation language should be as expressive as possible, or be efficiently reasoned over [5]. Expressivity refers to the ability of the language to directly capture the nuances of the target language, which in our case is the legal language. Being efficient to reason over refers to the ability to use existing and efficient tools to reason over formulae denoted in this language.

In the present paper, we take the former approach, since expressivity is essential to both an easy to use editor (section 2) and validation (section 3). At the same time, among our goals is to support efficient reasoning over formulae denoting legislations.

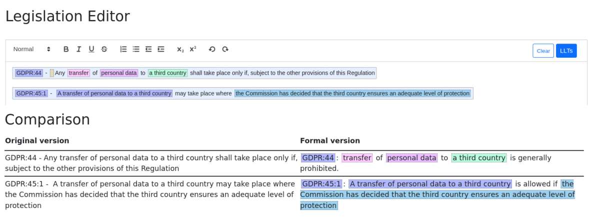
Our approach to deal with this apparent paradox is to put requirements on the expressive language. The main requirement is the ability to algorithmically translate formulae into a language in which efficient theorem proving is possible, which will be called the ”lower” language, as opposed to the ”higher”, expressive, language.

The LegAi lower language consists of classical first-order logic (see, e.g. [2]), to which we add a second, non-classical, negation operator and one ”Obligation” operator.

The LegAi higher language consists of vocabulary and Legal Linguistics Templates (LLTs).

Vocabulary and **LLTs** can be considered as logical terms and connectives, respectively. As an example, consider the vocabulary $personalData(DataSubject)$, which is

[1]See discussion in Sec. 3.2 .



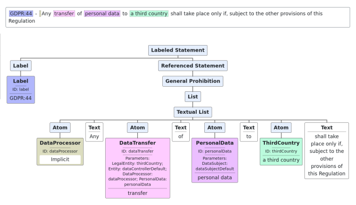
Legislation Editor

GDPR:44 - Any transfer of personal data to a third country shall take place only if, subject to the other provisions of this Regulation

GDPR:45:1 - A transfer of personal data to a third country may take place where the Commission has decided that the third country ensures an adequate level of protection

Comparison

Original version	Formal version
GDPR:44 - Any transfer of personal data to a third country shall take place only if, subject to the other provisions of this Regulation	GDPR:44 : transfer of personal data to a third country is generally prohibited.
GDPR:45:1 - A transfer of personal data to a third country may take place where the Commission has decided that the third country ensures an adequate level of protection	GDPR:45:1 : A transfer of personal data to a third country is allowed if the Commission has decided that the third country ensures an adequate level of protection



GDPR:44 - Any transfer of personal data to a third country shall take place only if, subject to the other provisions of this Regulation

a first-order term with one parameter, which is a free variable. An example of an LLT is $General - Prohibition(List)$, which accepts a list of other LLTs and denotes a legal prohibition, which can be overcome later via exceptions. Free variables are universally quantified over the whole legal knowledge base. So far, we have not seen a need to scope variable quantification locally.

Each LLT comes with its own ”semantics”, which are denoted in the lower language. The semantics are denoted recursively. For example, given the semantics operator $\mathscr{S}$:

$$\mathscr{S}\{General - Prohibition(List))\} = Obligation(\neg\mathscr{S}\{List\}) \tag{1}$$

3. The LegAi Annotation Editor

The goal of the LegAi annotation editor is to support annotating legal texts with the LegAi higher language, while at the same time, supporting validation of this annotations, which will be described in the next section.

The editor can be found online[2] and requires registration. After registration, an email to activate the account must be sent to the code maintainer, who associates the new user to a specific account with specific rights.

Once logged in, the user has the ability to paste legal texts into an annotation editor. The user starts with selecting a sentence they wish to annotate and then a (top-level) LLT wizard opens. The wizard guides the user in the annotation process, shows the possible/required LLTs that can/must be nested.

An example of annotating two simplified articles of the GDPR is shown on the top left of the figure above.

4. Ensuring Correctness via a Reverse Translation

In order to check if an annotation captures correctly the meaning of a legal sentence, the user can view the formal tree structure, as defined in the previous section and shown on the right of the figure above.

Nevertheless, such a formal structure is not easily readable by non-logicians. In order to rectify that, a feature called reverse translation was added to the editor.

The reverse translation, which is accessible on a separate tab, generates a legal text from the formal tree structure. It then places the reversely generated text next to the original text. The user can then compare the two versions and decide if the formal version correctly captures the meaning of the original one, as shown on the bottom left of the figure above.

[2] `https://legai.uni.lu`

5. Conclusion

In this paper we have presented the LegAi annotation editor. This editor is currently used by legal firms in order to generate legal knowledge bases for specific legal problems.

The editor is accompanied by the LegAi decision support tool (not described in this paper), which allows automated reasoning over the knowledge bases.

The LegAi higher language is currently translated only into the LegAi lower language. In the future, we plan on translating the higher language into TPTP [16] and other formats. The knowledge base, denoted in the higher language, can be exported to be used with other tools.

References

[1] Cesare Bartolini, Gabriele Lenzini, and Cristiana Santos. An interdisciplinary methodology to validate formal representations of legal text applied to the gdpr. In JURISIN, 2018.

[2] Barwise, Jon. ”An introduction to first-order logic.” Studies in Logic and the Foundations of Mathematics. Vol. 90. Elsevier, 5-46, 1977.

[3] Trevor JM Bench-Capon et al. Logic programming for large scale applications in law: A formalisation of supplementary benefit legislation. In Proceedings of ICAIL, pages 190–198. ACM, 1987

[4] Anne-Marie Burley and Walter Mattli. Europe before the court: a political theory of legal integration. International organization, 47(1):41–76, 1993.

[5] Benzmüller, Christoph. ”Universal (meta-) logical reasoning: Recent successes.” Science of Computer Programming 172 : 48-62, (2019).

[6] Mustafa Hashmi and Guido Governatori. Norms modeling constructs of business process compliance management frameworks: a conceptual evaluation. Artif. Intell. Law, 26(3):251–305, 2018

[7] Kowalski, Robert. ”Logical english.” Proceedings of Logic and Practice of Programming (LPOP), 2020.

[8] Libal, Tomer. ”Towards automated GDPR compliance checking.” International Workshop on the Foundations of Trustworthy AI Integrating Learning, Optimization and Reasoning. Springer, Cham, 2020.

[9] Libal, Tomer, and Matteo Pascucci. ”Automated reasoning in normative detachment structures with ideal conditions.” Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law. 2019.

[10] Martynas Mockus and Monica Palmirani. Legal ontology for open government data mashups. In 2017 Conference for E-Democracy and Open Government (CeDEM), pages 113–124. IEEE, 2017.

[11] Novotna, Tereza and Tomer Libal. ”An Evaluation of Methodologies for Legal Formalization”. Proceedings of the ”International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems. Springer, 2022.

[12] Monica Palmirani and Guido Governatori. Modelling legal knowledge for gdpr compliance checking. In Legal Knowledge and Information Systems: JURIX, volume 313, pages 101–110. IOS Press, 2018.

[13] Robaldo, Livio, et al. ”Formalizing GDPR provisions in reified I/O logic: the DAPRECO knowledge base.” Journal of Logic, Language and Information 29.4 : 401-449. 2020.

[14] Sergot, Marek J., et al. ”The British Nationality Act as a logic program.” Communications of the ACM 29.5 : 370-386, 1986.

[15] Soavi, Michele, et al. ”From Legal Contracts to Formal Specifications: A Systematic Literature Review.” SN Computer Science 3.5 : 1-25. 2022.

[16] Sutcliffe, Geoff. ”The TPTP problem library and associated infrastructure.” Journal of Automated Reasoning 43.4 : 337-362. 2009.

[17] Weaver, David A., and Bruce Bimber. ”Finding news stories: a comparison of searches using LexisNexis and Google News.” Journalism & Mass Communication Quarterly 85.3 515-530, 2008.