# Large-Scale Legal Reasoning with Rules and Databases

Grigoris Antoniou, George Baryannis, Sotiris Batsakis and Ilias Tachmazidis
*University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, UK*
{g.antoniou,g.bargiannis,s.batsakis,i.tachmazidis}@hud.ac.uk

Guido Governatori, Mohammad Badiul Islam and Qing Liu
*Data61, CSIRO, Dutton Park, QLD 4102, Brisbane, Australia*
{Guido.Governatori,badiul.islam,q.liu}@data61.csiro.au

Livio Robaldo
*Legal Innovation Lab Wales, Hillary Rodham Clinton School of Law, Swansea University, Singleton Park, Swansea, SA2 8PP, UK*
livio.robaldo@swansea.ac.uk

Giovanni Siragusa
*University of Turin, Via Pessinetto 12, 10149 Torino, Italy*
siragusa@di.unito.it

**Abstract**

Traditionally, computational knowledge representation and reasoning focused its attention on rich domains such as the law. The main underlying assumption of traditional legal knowledge representation and reasoning is that knowledge and data are both available in main memory. However, in the era of big data, where large amounts of data are generated daily, an increasing range of scientific disciplines, as well as business and human activities, are becoming data-driven. This chapter summarises existing research on legal representation and reasoning in order to uncover technical challenges associated both with the integration of rules and databases and with the main concepts of the big data landscape. We expect these challenges lead naturally to future research directions towards achieving large scale legal reasoning with rules and databases.

Vol. 8 No. 4 2021
Journal of Applied Logics — IfCoLog Journal of Logics and their Applications

# 1 Introduction

Since the emergence of computational knowledge representation and reasoning (KR), the domain of law has been a prime focus of attention as it is a rich domain full of explicit and implicit representation phenomena. From early Prolog-based approaches [67, 69] to elaborate logic-based mechanisms for dealing with, among others, notions of defeasibility, obligation and permission, the legal domain has been an inspiration for generations of KR researchers [4, 30, 50, 70].

Knowledge representation has been used to provide formal accounts of legal provisions and regulations, while reasoning has been used to facilitate legal decision support and compliance checking. Despite the variety of approaches used, they all share a common feature: the focus has always been on capturing elaborate knowledge phenomena while the data has always been small. As a consequence, one underlying assumption has been that all knowledge and data are available in main memory. This assumption has been reasonable until recently, but can be questioned with the emergence of *big data*. We now live in an era where unprecedented amounts of data become available through organisations, sensor networks and social media. An increasing range of scientific disciplines, as well as business and human activities, are becoming data-driven.

Since legislation is at the basis of and regulates our everyday life and societies, many examples of big data such as medical records in e-Health or financial data, must comply with, and are thus highly dependent on, specific norms. For instance, a sample database related to the US Food and Drug Administration (FDA) Adverse Event Reporting System (FAERS) contains over 3 million records to cover only the first quarter of 2014 [48]. Any standard reasoning system would reach its limits if data over longer periods of time need to be audited.

Another source of huge amounts of data related to law is the financial domain, in which millions of transactions take place every single day and are subject to regulation on, among others, taxation, anti money laundering, consumer rights and data protection. While data mining is being used in the financial domain, it is arguably an area that would benefit from legal reasoning directly related to relevant legislation. This might indicatively entail checking for and ensuring compliance with reporting requirements, or traversing across financial transaction databases to check for potential violations of legislations.

Similarly, building applications and property/site development are covered by a variety of local and national laws and regulations. To develop and assess relevant applications, it may be necessary to consider the legal requirements in conjunction with geodata relating to morphology of the site and its surroundings, use of space and so on.

Industries in the aforementioned and other domains are feeling increasingly overwhelmed with the expanding set of legislation and case law available in recent years, as a consequence of the global financial crisis, among others. Consider, for example, the European Union active legislation, which was estimated to be 170,000 pages long in 2005 and is expected to reach 351,000 pages by 2020 assuming that legislation trends continue at the same rate [54]. As the law becomes more complex, conflicting and ever-changing, more advanced methodologies are required for analysing, representing and reasoning on legal knowledge.

While, the term "big data" is usually associated with machine learning, we argue that particularly in law there is also a need for symbolic approaches. Legal provisions and regulations are considered as being formal and legal decision making requires clear references to them. Stated another way, in the legal domain there is also a need for *explainable artificial intelligence*, as it has always been done in legal reasoning.

So what are the implications of this big data era on legal reasoning? On the one hand, as already explained above, a combination of legal reasoning with big data opens up new opportunities to provide legal decision support and compliance checking in an enhanced set of applications. On the other hand, there are new technical challenges that need to be addressed when faced with big data:

- Rules and data integration: while big data is stored in databases of various forms, reasoning is often performed using rule engines. Integrated solutions are necessary so that rule engines can seamlessly access and reason with big data in large scale databases.
- Volume: When the amount of data is huge, one cannot assume that all data is available in main memory. Hence, any approach that relies on this assumption needs to be adapted in order to work on larger scales.
- Velocity: In applications where one wishes to perform decision making close to the time data is generated, the dynamicity of data needs to be taken into account.
- Variety: In many applications, there is a need for a uniform manner of accessing and reasoning with data from disparate, heterogeneous sources, following different formats and structures.

The aim of this chapter is to present the state of the art in legal reasoning with rules and databases and explore the challenges faced by existing approaches when moving to larger scales and when integrating rule-based and database systems. In doing so, the chapter aims to stimulate the evolution of the area of legal reasoning so that it becomes more relevant in the new data-driven era.

The remainder of this chapter is organised as follows. Section 2 provides an overview of previous research in legal representation and reasoning. Section 3 discusses the application of legal reasoning in practice, first dealing with case studies of increasing scale, then discussing the integration of rules and databases and a possible solution through the RuleRS system. Then, Section 4 provides a description of technical challenges arising both from the integration of rules and databases and large scale case studies. Finally, Section 5 summarises findings and briefly discusses their importance.

# 2 Legal Representation and Reasoning Approaches

## 2.1 Rule-based Approaches

A quite significant subset of legal representation and reasoning approaches relies on logic-based representation and rule-based reasoning. The benefits of rule-based approaches stem mainly from their naturalness, which facilitates comprehension of the represented knowledge [52]. Rules, representing domain knowledge, are normally in the "IF conditions THEN conclusion" form; in the legal domain, conditions are the norms and consequence is the legal effect. To apply rule-based reasoning in the legal domain, the meaning of legal texts needs to be interpreted and modelled, in order to transform the legal norms to logical rules for permitting reasoning [21].

According to Negnevitsky [56], the main advantages of rule-based approaches are:

- compact representation of general knowledge,
- natural knowledge representation in the form of if-then rules that reflect the problem-solving procedure explained by the domain experts,
- modularity of structure where each rule is an independent piece of knowledge
- separation of knowledge from its process,
- justification of the determinations by explaining how the system arrived at a particular conclusion and by providing audit trails.

There are, however, a number of issues that pertain to the knowledge acquisition bottleneck, or inference efficiency, especially for large scale reasoning. Sections 2.2 to 2.4 summarise the most important rule-based legal reasoning approaches.

## 2.2 Early Logic-based Approaches

The earliest well-established approach to rule-based legal reasoning involved the use of subsets of first-order logic for knowledge representation and Prolog-based reasoning. The most prominent example is Sergot et al.'s seminal work on the British Nationality Act [69], where the authors expressed legal knowledge in the form of extended Horn logic programs that allow negation as failure. The authors present an excellent account of the intricacies of encoding actual legislation as rules, especially with regard to the treatment of negation and cases where double negation is introduced.

Subsequent work [49] focused, among others, on the encoding of exceptions within a particular legislation, representing them explicitly by negative conditions in the rules. While this is suitable for self-contained and stable legislation, it may require some level of rewriting whenever previously unknown exceptions (or chains of exceptions) are introduced or discovered. Moreover, in both of these works deontic concepts such as permission or obligation which are a common occurrence in legislation, have to be represented explicitly within predicate names. This is an expected characteristic when legal knowledge representation relies on standard predicate logic [11].

## 2.3 Description Logic-based Approaches

Following the advent of the Semantic Web, several research efforts focused on examining whether description logics and ontologies are suitable candidates for representing and reasoning about legislation (see [3] and [57], among others). An ontology is defined as a formal, explicit specification of a shared conceptualisation [72]. The reusability and sharing features of ontologies are of critical importance to the legal reasoning domain, due to the complexity involved in legal documents. This complexity can be viewed from two different perspectives [31]:

- the complexity of the language used in legal documents, due to, among others, the open texture property and the incomplete definition of many legal concepts [29].
- the complexity brought on by the amount of information that must be collected and processed in order for lawyers or judges to evaluate a case and litigation to proceed [76].

A prime example of legal reasoning approaches using description logics is HARNESS [74] (also known as OWL Judge [75]), which shows that well-established sound

and decidable description logic reasoners such as Pellet can be exploited for legal reasoning, if, however, a significant compromise in terms of expressiveness is made. The most important issue is that relationships can only be expressed between concepts and not between individuals: for instance, as exemplified in Van de Ven et al. (2008), if we have statements expressing the facts that a donor owns a copyright donation and that a donor retains some rights, there is no way to express (in pure OWL) that the donor in both cases is the same individual. This can be expressed via rules (e.g. written in SWRL); however, to retain decidability these rules must be restricted to a so-called DL-safe subset [58].

Description logics provide an alternative formalisation to classical logic but still face similar issues with regard to the treatment of negation and the encoding of deontic notions. The issues related to negation are due to the fact that both classical and description logics are monotonic: logical consequences cannot be retracted, once entailed. However, the nature of law requires legal consequences to adapt in light of new evidence; any conflicts between different regulations must be accounted for and resolved [11].

## 2.4 Defeasible and Deontic Logic-based Approaches

The aforementioned issues led researchers to employ non-monotonic logic for the purposes of legal reasoning. An example is the Defeasible Logic framework [6], where rules can either behave in the classical sense (*strict*), they can be defeated by contrary evidence (*defeasible*), or they can be used only to prevent conclusions (*defeaters*). Defeasible Logic has been successfully used for legal reasoning applications [7, 33, 40, 35] and it has been proven that other formalisms used successfully for legal reasoning correspond to variants of Defeasible Logic [34]).

As already mentioned, the notions of permission and obligation are inherent in legal reasoning but are not explicitly defined in any of the logic systems described so far; deontic logic was introduced to serve this purpose. As formalised in [43], permission and obligation are represented by modal operators and are connected to each other through axioms and inference rules. While there has been some philosophical criticism on deontic logic due to its admission of several paradoxes (e.g. the gentle murderer), deontic modalities have been introduced to various logics to make them more suitable for reasoning with legal norms. Sergot [68] uses a combination of deontic logic and the notions of action and agents to be able to derive all possible normative positions (e.g. right, duty, privilege) and assist in policy and contract negotiation. A similar proposal [65, 10] uses reified Input/Output logic [66] to formalise the EU General Data Protection Regulation (GDPR) in 966 if-then rules.

Defeasible Deontic Logic [38, 41] is the result of integrating deontic notions (beliefs, intentions, obligations and permissions) to the aforementioned Defeasible Logic framework. Defeasible Deontic Logic has been successfully used for applications in legal reasoning and it is has been shown that it does not suffer from problems affecting other logics used for reasoning about norms and compliance [36, 35, 48]. Thus, Defeasible Deontic Logic is a conceptually sound approach for the representation of regulations and at the same time, it offers a computationally feasible environment to reason about them [38].

### 2.5 Case-based Approaches

Apart from rule-based approaches, a number of different solutions have been proposed for representation and reasoning in the legal domain. These are summarised next. This section discusses case-based approaches, followed by case-rule hybrids (Section 2.6) and argumentation-based approaches (Section 2.7).

Rule-based legal reasoning approaches are more suited to legal systems that are primarily based on civil law, due to their inherent rule-based nature and the fact they focus on conflicts arising from conflicting norms and not from interpretation [14]. On the other hand, common law places precedents at the center of normative reasoning, which makes case-based approaches more applicable. Case-based representations store a large set of previous cases with their solutions in the case base (or case library) and use them whenever a similar new case has to be dealt with. The case-based system performs inference in four phases known as the CBR cycle [2]): retrieve, reuse, revise and retain. Quite often, the solution contained in the retrieved case(s) is adapted to meet the requirements of the new case.

An important advantage of case-based representation is its ability to express specialized knowledge. This allows them to circumvent interpretation problems suffered by rules (due to their generality). Also, knowledge acquisition may be slightly easier than rule-based approaches, due to the availability of cases in most application domains. However, case-based approaches face a number of issues such as the inability to express general knowledge, poor explanations and inference inefficiency, especially for larger case bases [62].

The most prominent examples of case-based legal reasoning are HYPO [9], CATO [5] and GREBE [17]. HYPO represents cases in the form of dimensions which determine the degree of commonality between two precedent cases: a precedent is more "on-point", if it shares more dimensions with the case at hand than another. CATO replaces dimensions with boolean factors organised in a hierarchy. GREBE is actually a rule/case hybrid, since reasoning relies on any combination of rules modeling legislation and cases represented using semantic networks (a pre-

cursor to ontologies in the Semantic Web). As noted in [13], using dimensions or factors to determine legal consequences is relatively tractable, but the initial step of extracting these dimensions or factors from case facts is deeply problematic.

## 2.6 Hybrid Approaches

A number of attempts have been made to integrate rule-based and case-based representations [62]. Since rules represent general knowledge of the domain, whereas cases encompass specific knowledge gained from experience, the combination of both approaches turns out to be natural and useful.

In legal reasoning, such hybrid solutions are capable of addressing issues arising due to the existence of "open-textured" (i.e., not well defined and imprecise) rule terms or unstated prerequisite conditions and exceptions or circularities in rule definitions [64]. Examples of hybrid legal representation and reasoning systems are CABARET [64], DANIEL [20], GREBE [18, 16], and SHYSTER-MYCIN [1].

## 2.7 Argumentation-based Approaches

Regardless of the legal system applied, legal reasoning at its core is a process of argumentation, with opposing sides attempting to justify their own interpretation. As succinctly stated in [61], legal reasoning goes beyond the literal meaning of rules and involves appeals to precedent, principle, policy and purpose, as well as the construction of and attack on arguments. AI and law research has addressed this with models that are based on Dung's influential work on argumentation frameworks [26]. A notable example is Carneades [32], a model and a system for constructing and evaluating arguments that has been applied in a legal context. Using Carneades, one can apply pre-specified argument schemes that rely on established proof standards such as "clear and convincing evidence" or "beyond reasonable doubt".

ASPIC+ [60] takes a more generic approach, providing a means of producing argumentation frameworks tailored to different needs in terms of the structure of arguments, the nature of attacks and the use of preferences. However, neither Carneades nor any ASPIC+ framework can be used as-is for legal reasoning: they need to be instantiated using a logic language. For instance, versions of Carneades have used Constraint Handling Rules to represent argumentation schemes, while any ASPIC+ framework can be instantiated using a language that can model strict and defeasible rules, such as those in the previously mentioned Defeasible Logic framework.

# 3 Legal Reasoning with Rules and Databases in Practice

As detailed in the previous section, researchers have proposed a multitude of different approaches to legal representation and reasoning, each with their own advantages and disadvantages. Focusing on rule-based approaches specifically, regardless of their individual characteristics, two major issues have not yet been adequately addressed, to the best of our knowledge. These involve handling significantly large datasets and achieving efficient integration between legal rules and databases. In this section, we explore how current rule-based legal reasoning approaches fare in relation to these issues.

## 3.1 Exploring Case Studies of Different Scale

As part of the MIREL project[1], practical legal reasoning applications were explored to complement theoretical analysis. For instance, in [11], several legal reasoning approaches were applied on real-world use cases. The approaches examined included answer set programming (ASP) [19], defeasible logic and ASPIC+-based argumentation. The use cases involved the presumption of innocence axioms, blockchain-based contracts use case and the FDA Adverse Event Reporting System.

The first use case (presumption of innocence) involves only a few rules but demonstrates the importance of semantics and how different formalisms deal with conflicting facts and rules, especially in the case of missing preferences between rules. The second use case is an example of rules within a contract, and is interesting due to including notions of permission, obligation and reparation. The third use case involves part of the rules applied in the FDA reporting system mentioned in the introduction. Since the number of rules and cases is big, the third use case is very relevant to the challenges of large scale reasoning. More details on the three use cases and example implementations in the three formalisms (ASP, defeasible logic and argumentation) can be found in [11].

The three formalisms were selected because of their support for complex rules involving conflicts and priorities, as is typically the case of legal reasoning, and the availability of stable tools for reasoning. All three formalisms were expressive enough for representing rules involved in the three use cases, but the user must be familiar with the underlying semantics, since in some cases the rules must be modified accordingly in order to achieve the desired behaviour. But besides their differences, the three approaches can form the basis of a large scale reasoning implementation.

[1] `https://www.mirelproject.eu`

The advantage of ASP is its expressiveness since it offers support for disjunction, strong negation and negation as failure and additional constructs such as aggregation functions; in contrast, argumentation has significantly restricted expressiveness. In terms of computational complexity, defeasible logic offers reasoning with lower complexity, in general. Overall, defeasible logic seems to provide the best trade-off between expressiveness and complexity. ASP and argumentation do not offer out of the box support for deontic operators. These have to be added by explicitly including additional expressions (predicates or propositions) and rules formalising the logical relationships between the new predicates' various instances. The need to include rules to simulate the underlying semantics can result in a large number of rules, where the rules for the implementation of the logic exceed by far the rules corresponding to the legal document to be encoded by the rule-base. In general, most of the additional rules are not used; this means that encoders can use their domain knowledge and expertise to decide which rules are needed and ignored. However, this process is time-consuming and error-prone and might make the representation not suitable for some applications.

The most complex use case in [11], a subset of FDA Adverse Event Reporting System, when implemented contains approximately 100 rules for all three formalisms. Reasoning times for three formalisms did not exceed a few seconds. This means that reasoning is efficient for hundreds of rules, but challenges may arise for even larger rule sets or in case reasoning results in one rule set depend on completing reasoning on another set. One potential bottleneck identified is representation, since manual encoding of rules and case related facts may be time consuming and may require expertise in the formalism used for reasoning. However, recent experiments on encoding pieces of legislation in Defeasible Logic [42, 46] indicate that legal domain experts can represent (large) legal documents in this logic with minimal training in the language of defeasible deontic logic and its semantics with no previous experience in formal logic, argumentation or logic programming.

### 3.2 Integration Between Legal Rules and Databases

For many applications, necessary data is stored in (relational) databases. Various organizations may use the data from existing databases to comply with various regulations and guidelines, take decisions and create reports based on regulations (and other normative and legislative documents). For example, Australian financial institutions are subject to Financial Sector (Collection of Data) Act 2001, with regard to what (financial) information to report to the relevant regulators (e.g., Australian Prudential Regulator Authority); government departments and agencies are required to comply with the Public Governance Performance and Accountability

Act 2013 and Public Governance Performance and Accountability Rule 2014 for their annual financial reporting. The requirements about what, when and in what forms to comply (and related exceptions) are given in the (relevant) regulations while the (financial and other) data is stored in the databases of the institutions that have to generate reports about the data using legal reasoning.

Accordingly, in these scenarios, one has to perform some legal reasoning (for example to understand what are the actual requirements that apply in a given case) based on the information stored in enterprise databases. In fact, legal reasoning consists of five elements which lead to a decision that can be decided as either accepted or rejected [2]. The components are: issues or cases (legal), rules, facts, analysis and conclusion. The argument for a particular issue has to align with the legal rule and relevant facts corresponding to the rule. Overall, the process is analysed and apply the facts from database to the rules for generating a conclusion. Consequently, the facts stored in the enterprise database are required to apply the rules and perform legal reasoning.

Typically, database management systems involve a relatively small number of relations or files holding a large number of records, whereas rule-based systems consist of a large number of relationships with a small number of records [63]. Additionally, relational databases essentially represent knowledge in a first-order logic formalism and query languages mostly exploit first-order logic features. However, as detailed in Section 2, first-order logic is not fully suitable to represent legal knowledge. This means that in general, we cannot use solely database queries, but we have to integrate the information stored in a database with rule systems specialised in legal reasoning.

A possible solution to integrating rules with databases would be to encode and store rules in a separate application program and then align with databases. However, in this manner, it would often be difficult to adapt the program if regulations change. Additionally, it could not be guaranteed that databases and rule-based systems are consistently amended. Another solution would be to couple databases with an expert system, but this would not solve the consistency problem since data is in one system, and the rules are in another one [71]. Stonebraker suggests that rule systems integrated into the (relational) database system could be the possible solution. In this circumstance, it is required to integrate a database to serve legal obligations since traditional database architecture is not capable of reporting regulatory requirements.

[2] `https://groups.csail.mit.edu/dig/TAMI/inprogress/LegalReasoning.html`

### 3.3 RuleRS: A Solution to the Integration Problem

This section demonstrates RuleRS [48], a possible solution where rules and databases are integrated. Initially, we are focusing on the mapping between the two vocabularies representing rules and databases. The fundamental idea behind the mapping is that data stored in the database correspond to facts in a defeasible theory and these facts can be retrieved from the database using queries (SQL, JSON). Thus, each fact corresponds to a query and a mapping is a statement that can be true or false depending on the value of its arguments/variables.

The *RuleRS* design architecture, shown in Figure 1, consists of five main system components. In particular, the key system components of RuleRS are: 1) I/O Interface, 2) Database facts 3) Formal Rules, 4) Predicates, and 5) Rule engine (SPINdle Reasoner). The following subsections provide a short outline of the RuleRS internal components and their functions.

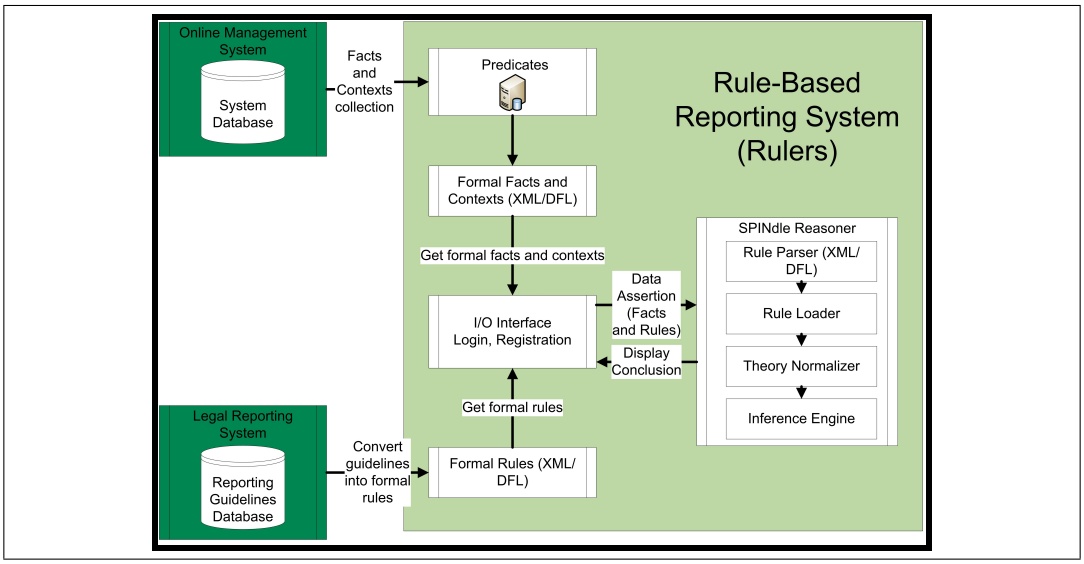


Figure 1: Rule-based Reporting System (RuleRS)

#### 3.3.1 I/O Interface

The I/O Interface is implemented in Java to bridge RuleRS components and interacting with each other. The I/O interface is used to query data predicates (SQL or JSON files) and to generate facts and contexts in formal notation in Defeasible Logic syntax, and the rule engine (SPINdle reasoner) receives this as a parameter. The

I/O interface also displays the final remarks or comments for each of the incidents and predicates.

### 3.3.2 Database Facts

This section describes how to obtain facts from databases. In RuleRS, facts can be true or false for specific information from the database which is mapped with the literals rules. We have used either SQL or JSON (JavaScript Object Notation) syntax[3] syntax (or a combination of them) to represent database facts. Each of the facts is generated by querying the database and is sent to the reasoner for further processing.

### 3.3.3 Formal Rule Base

One of the prominent features of the RuleRS system is its ability to perform reasoning based on legal requirements. As we alluded to in the introduction, such regulatory requirements are represented as formal rules in Defeasible Deontic Logic [38, 41]. To enable their use with the rule engine used by RuleRS (SPINdle, see the next section) the rules are stored in the DFL format [51]. At this stage, the rules are created manually and (semi-)automatically by legal knowledge engineers and stored in a knowledge base.

### 3.3.4 Predicates

As specified earlier, since there is no direct correspondence between the literals encoding rules and the table/attributes of the database schema, we have to establish a mapping among them to enable the integration of rules and instances in the database. We named this mapping "predicates". The fundamental idea behind predicates is that data stored in the database correspond to facts in a defeasible theory and these facts can be retrieved from the database using queries. Thus, each fact corresponds to an SQL/JSON query and a predicate is a statement that can be true or false depending on the value of its arguments/variables. A predicate with $n$ arguments is an $n - ary$ relation mapping literals and a set of attributes. A predicate in RuleRS corresponds to a database view, i.e.; a named query, where the name is literal to be used by the defeasible rules. The details are the query to be run to determine if the predicate is true or false for a given set of parameters. In case the output of the query is not empty, the predicate is true and is passed to the defeasible theory as fact.

[3] http://json.org

In RuleRS, predicate consists of two components: (1) predicate name and (2) predicate details. *Predicate name* represents the action(s), condition(s) or indisputable statement(s), and passed on to the rule engine, SPINdle as defeasible fact (literal and modal literal) [37, 38, 39] or actions that have been performed. For example, the fact "There is a risk for an incident" is represented by "$riskForIncident$" and passed as "$>> riskForIncident$" to SPINdle if it is returned as true from the relational database. "Predicate details" include the "incident details" and may be stored as an SQL statement or converted to JSON to create a bridge between the data stored in the database and the terms passed as predicates (input case) to the rule engine. The SQL or JSON statements can be created in the initialisation of RuleRS with all of the incidents along with all of the predicates for each of the incidents or dynamically add it later.

Incident ID and relevant details of the incidents are also included for each of the predicates and predicates names include relevant incident information such as "riskForIncident.sql" (for SQL statement) or "riskForIncident.json" (for JSON Statement). The following snippet illustrates the SQL syntax adopted by RuleRS for the example of the "riskForIncident" predicate:

```
SELECT incidentID, IncidentDetails, IncidentDetails1,
IncidentDetails2  FROM tblIncident
WHERE incidentID='XXXXXX'
```

In this example, IncidentDetails, IncidentDetails1, IncidentDetails2 are substituted for the place- holders in the "riskForIncident" predicate from relational databases for the incidentID 'XXXXXX. Using JSON, the syntax for the "riskForIncident" predicate is:

```
{"riskForIncident":
{ "incidentID":"XXXXXX",
"IncidentDetails":"ABC",
"IncidentDetails1":null,
"IncidentDetails2":"XYZ"}}
```

In the next step, the records and incidents for which there is a match in the relational database are transformed into predicates to be used by the SPINdle rule engine [51], and forwarded to SPINdle for further processing using the I/O interface to make the process dynamic.

### 3.3.5 The Rule Reasoner

RuleRS uses SPINdle Reasoner [4] [51], a Java-based implementation of Defeasible Logic that computes the extension of a defeasible theory. SPINdle supports Modal Defeasible Logic and all types of Defeasible Logic rule, such as facts, strict rules, defeasible rules, defeaters, and superiority. In summary, SPINdle is a powerful tool which accepts rules, facts, monotonic and non-monotonic (modal) rules for reasoning with inconsistent and incomplete information. In RuleRS, SPINdle Reasoner receives the formal facts, contexts as predicates from predicate file generated for data stored in the associated relational databases and computes definite or defeasible inferences which are then displayed by the I/O interface.

# 4 Challenges and Future Research Directions

A number of different challenges arise when attempting to move towards large scale legal reasoning with rules and databases. Some of these challenges are directly related to the integration between rules and data and are discussed in Section 4.1. Others are linked to issues raised by large scale data and are discussed in Section 4.2.

## 4.1 Integration between Rules and Databases

Stonebraker [71] discusses three possible forms that bring rules with database systems:

- rule policy can be written down in a booklet and distributed to people,
- the rules can reside in an application program which accesses the databases,
- a knowledge base can reside inside the DBMS by which we can guarantee that the data is consistent with the rules

The author expected that the last form will be the one to be adopted as a major approach. However, we argue that the last form may work well for a single database with small amount of rules but poses some significant challenges for large scale legal reasoning. A number of challenges are raised when attempting to integrate rules and databases, especially at larger scales and these are detailed next.

[4] SPINDle Reasoner is available to download freely from `http://spin.nicta.org.au/spindle/tools.html` under LGPL license agreement (`https://opensource.org/licenses/lgpl-license`)

### 4.1.1 Common languages

The values encoding regulation and guidelines (legal documents) and the databases (schemas) used in conjunction with the rules are in general developed independently and are likely to have different vocabulary. This may lead to "Tower of Babel" issues, due to the absence of "common languages" between regulations and databases. There is no direct correspondence between the literals used by the rules and the table/attributes of the database schema. Accordingly, we have to establish a mapping between them to enable the integration of rules and instances in the database. The connection between rules and databases is demonstrated by a number of systems [48, 47, 24].

### 4.1.2 Integrating varieties of data sources with rule engines

Another challenge involves the integration of data coming from disparate sources with rule engines. Each source could publish data in their own format and all of these formats would need to be brought together to construct schema-based conditions for rules. This is quite a cumbersome task for knowledge engineering. Furthermore, when database schemas or rules change, schema-based indices will also be affected due to the strong coupling.

### 4.1.3 Inference efficiency

In the case of defeasible deontic logic in legal domain, each condition in a rule could be represented by a complex query that involves multiple selections, projections and joins across multiple tables and databases. Existing schema-based index approaches cannot address this complexity well. Furthermore, rules in the legal domain have not only dependent relationship but also defeater relationship. Together with issues such as reparation chain handling, they bring more dynamics during reasoning process which places even heavier burden to inference engines.

### 4.1.4 Reactive inference

The existing reasoning process in systems such as RuleRS is that the inference engine looks for rules which match facts stored in the working memory or provided by users. One rule is selected from the "conflict set" and executed to generate a new fact. Then the inference engine will continue the reasoning based on the new fact together with the previous given facts. We call this as reactive inference because the inference engine only reasons based on what is given but does not interact with databases to seek "unknown" facts proactively. Proactive inference is critically important

when it is highly unlikely for users to know all facts beforehand. Furthermore, the assumption of storing facts "in memory" does not hold for large scale reasoning, as detailed in Section 4.2.2.

### 4.1.5 Rules as data

Rules could be treated as data and stored in database systems, to make it easier for the rules to be triggered and executed as and when required [59]. The main issue with storing rules in the database is that the database is not capable of handling deontic concepts. To correctly model the provision corresponding to prescriptive norms, we have to supplement the language with deontic operators, and the databases are not capable of handling these specific features.

Rules treated as data could create further challenges. Legal reasoning integrating rules and databases are not limited to any particular regulations. Hence, the database could be aligned to one-to-many regulations, establishing n-ary relations among these. If such rules are treated as data and stored in databases, then the task of amending them if necessary becomes even harder, since each of the rules could connect with another rule leading to nested and correlated queries. Such queries are usually avoided due to their complexity [5]. Query maintainability and filtering also create further challenges.

## 4.2 Large-Scale Legal Reasoning

### 4.2.1 Representation

As discussed in Sections 2 and 3.1, there are several formalisms that can be used for representing legal norms and facts about cases, such as answer set programming, argumentation and defeasible logic. Although such formalisms are expressive enough for representing legal rules and efficient reasoning mechanisms and tools exist for them, encoding the rules may, in some cases, turn out to be a complex and time-consuming process, since the representation of a legal document can easily require thousand propositions and many thousand rules [42, 46].

In larger scales, the encoding process may face severe scalability issues and turn out to be a potential bottleneck for efficient large scale reasoning. Automating this process with the help of efficient natural language processing tools is an open research problem; there are several examples of preliminary results in literature [25, 28, 15, 77, 55].

[5] `http://www.sqlservice.se/sql-server-performance-death-by-correlated-subqueries/`

### 4.2.2 Volume

Traditional legal reasoning has been focused on storing and processing data in main memory over a single processor. This approach is indeed applicable to small legal documents. However, there is a limit on how many records an in-memory system can hold. In addition, utilising a single processor can lead to excessive processing time.

RuleRS [48] indicates that data can be processed record by record, namely querying the database and performing reasoning for each record separately. Experimental evaluation shows that this approach can evaluate each record within seconds on a standard laptop computer. They further estimate that auditing the data collected in the underlying database for a quarter, amounting to approximately 3 millions records, would require an estimated time of 8 hours examining record by record sequentially. It should be noted that these experiments followed a brute force approach. Clearly, given the fact that FAERS data that is readily available is already 10 times larger compared to the ones processed in [48], batch processing would require processing times in the order of days processing records one-by-one sequentially, unless there is a significant increase in available computational power. However, standard database optimisation techniques such as query grouping, optimisation, use of cursors, parallelising queries and reasoning and custom indexing can reduce this time significantly [53].

A record-by-record processing approach cannot be guaranteed for any given application. Thus, in other applications where all records need to be loaded and processed together, main memory would be a hard constraint considering applicability, when commodity hardware is used. Addressing memory constraints through the use of specialised hardware containing terabytes of memory should be coupled with a respective increase in the number of available processors in order to minimize processing time.

Recent advances in mass parallelisation could potentially address the limitations related to memory and processing time. It has been shown in literature [8, 22, 78] that mass parallelisation can be applied to various types of reasoning. Both supercomputers (e.g. a single large machine with hundreds of processors and a large shared main memory) and distributed settings (e.g. a large number of combined commodity machines that collectively provide multiple processors and a large distributed memory) can be used in order to speed up data processing. The advantages are twofold, since mass parallelisation: (a) could significantly reduce processing time as multiple cores can be used simultaneously, and (b) virtually alleviates the restriction on main memory as more memory can be easily added to the system.

### 4.2.3 Velocity

Financial transactions could potentially require real-time monitoring of day-to-day activity. Such functionality would depend on processing large amounts of transactions within seconds. For cases where reasoning needs to take place during a short window of time, close to the time that events take place, batch reasoning is no longer a viable solution. A prominent challenge in this situation is the efficient combination of streaming data with existing legal knowledge (e.g. applicable laws and past cases), essentially updating the latter. Stream reasoning has been studied in literature [44, 73], showing that only relatively simple rules could allow high throughput. In general, stream processing is intended for use cases where data is processed towards a single direction. However, in stream reasoning, recursive rules (i.e. rules that lead to inference loops) may lead to performance bottlenecks. In addition, within such a dynamic environment, incoming data could potentially invalidate previously asserted knowledge leading to a new set of knowledge, which would in turn change the set of conclusions. Recent advances in stream reasoning could provide a solution to this challenge, through systems such as ASTRO [23] and LARS [12, 27].

### 4.2.4 Variety

One of the main challenges in large-scale legal reasoning could be the integration of data coming from disparate sources. Each source could publish data in any possible format, ranging from images of scanned pages to machine-processable files. Thus, the first challenge is to translate all available data into machine processable data that can be readily stored and retrieved. Once this data transformation is achieved managing data that are stored in different formats (e.g. plain text, JSON, XML, RDF) would complicate legal reasoning as all data would need to be translated into a single format in order to have a uniform set of facts. Thus, in order to tackle data variety, all available data would need to be stored in a uniform format that would allow automated translation into facts of the chosen legal reasoning framework.

Existing work on semantic technologies can be used to address these challenges. Through the use of upper ontologies that provide definitions for a wide range of concepts, specialised legal ontologies such as LKIF [45] or bespoke ontologies, it can be ensured that all available data sources related to a large scale legal reasoning effort are eventually mapped into a unified body of knowledge.

## 5 Conclusion

This chapter argued that there is scope for research in AI and law with regard to performing effective legal reasoning when the associated knowledge and data is on a large scale and there is also a need for integration between rules and databases. A number of potential scenarios were discussed where this kind of reasoning would be useful, with use cases ranging from the pharmaceutical and financial to property development sectors.

Through a summary of state of the art and an analysis of applying rule-based legal reasoning and integrating rules and databases in practice, it becomes evident that current approaches are not fully equipped to handle large scale legal reasoning with rules and databases and face several challenges.

With regard to the problem of integration between rules and databases, the identified challenges relate to: (a) common languages; (b) integrating rule engines with various data sources; (c) inference efficiency; (d) reactive inference; and (e) rules as data. Additional challenges are encountered when moving towards larger scales, dealing with: (a) representation; (b) volume; (c) velocity; and (d) variety.

It is envisioned that these challenges, among others, will drive research on legal representation and reasoning in the near future, providing researchers at the confluence of AI and law with a multitude of potential avenues of investigation. By addressing some of these challenges, efficient, effective and successful large scale legal reasoning with rules and databases will be achievable in the era of big data.

## References

[1] Thomas A O'Callaghan, James Popple, and Eric McCreath. Shyster-mycin: A hybrid legal expert system. In *Proceedings of the Ninth International Conference on Artificial Intelligence and Law (ICAIL-03)*, pages 103–4, 06 2003. ISBN 1 58113 747 8. doi: 10.1145/1047788.1047814.

[2] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications. IOS Press*, 7:1:39–59, 1994.

[3] G. Ajani, G. Boella, L. Di Caro, L. Robaldo, L. Humphreys, S. Praduroux, P. Rossi, and A. Violato. The European legal taxonomy syllabus: A multi-lingual, multi-level ontology framework to untangle the web of European legal terminology. *Applied Ontology*, 2 (4), 2017.

[4] Marco Alberti, Federico Chesani, Marco Gavanelli, Evelina Lamma, Paola Mello, and Paolo Torroni. Verifiable Agent Interaction in Abductive Logic Programming: The SCIFF Framework. *ACM Trans. Comput. Logic*, 9(4):29:1–29:43, 2008. ISSN 1529-3785.

[5] Vincent Aleven. Using background knowledge in case-based legal reasoning: A computational model and an intelligent learning environment. *Artif. Intell.*, 150(1-2):183–237, 2003.

[6] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J. Maher. A Flexible Framework for Defeasible Logics. In Henry A. Kautz and Bruce W. Porter, editors, *AAAI/IAAI*, pages 405–410. AAAI Press / The MIT Press, 2000. ISBN 0-262-51112-6.

[7] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J Maher. Representation results for defeasible logic. *ACM Transactions on Computational Logic*, 2(2):255–287, 2001.

[8] Grigoris Antoniou, Sotiris Batsakis, Raghava Mutharaju, Jeff Z. Pan, Guilin Qi, Ilias Tachmazidis, Jacopo Urbani, and Zhangquan Zhou. A survey of large-scale reasoning on the web of data. *Knowledge Eng. Review*, 33:e21, 2018. doi: 10.1017/S0269888918000255.

[9] Kevin D. Ashley. *Modeling Legal Argument: Reasoning With Cases and Hypotheticals*. The Bradford Books, MIT Press, 1990.

[10] Cesare Bartolini, Andra Giurgiu, Gabriele Lenzini, and Livio Robaldo. Towards legal compliance by correlating standards and laws with a semi-automated methodology. In Tibor Bosse and Bert Bredeweg, editors, *BNAIC 2016: Artificial Intelligence - 28th Benelux Conference on Artificial Intelligence, Amsterdam, The Netherlands, November 10-11, 2016, Revised Selected Papers*, volume 765, pages 47–62. Springer, 2016.

[11] Sotiris Batsakis, George Baryannis, Guido Governatori, Ilias Tachmazidis, and Grigoris Antoniou. Legal Representation and Reasoning in Practice: A Critical Comparison. In *Legal Knowledge and Information Systems - JURIX 2018: The Thirty-first Annual Conference, Groningen, The Netherlands, 12-14 December 2018.*, pages 31–40, 2018. URL `https://doi.org/10.3233/978-1-61499-935-5-31`.

[12] Harald Beck, Minh Dao-Tran, and Thomas Eiter. Lars: A logic-based framework for analytic reasoning over streams. *Artif. Intell.*, 261:16–70, 2018. URL http://dblp.uni-trier.de/db/journals/ai/ai261.html#BeckDE18.

[13] Trevor J. M. Bench-Capon. What Makes a System a Legal Expert? In Burkhard SchÃdfer, editor, *JURIX*, volume 250 of *Frontiers in Artificial Intelligence and Applications*, pages 11–20. IOS Press, 2012. ISBN 978-1-61499-166-3.

[14] Trevor J. M. Bench-Capon and Henry Prakken. Introducing the Logic and Law Corner. *J. Log. Comput.*, 18(1):1–12, 2008.

[15] Guido Boella, Luigi Di Caro, Daniele Rispoli, and Livio Robaldo. A system for classifying multi-label text into eurovoc. In Enrico Francesconi and Bart Verheij, editors, *International Conference on Artificial Intelligence and Law, ICAIL '13, Rome, Italy, June 10-14, 2013*, pages 239–240. ACM, 2013.

[16] Karl Branting. A reduction-graph model of precedent in legal analysis. *Artif. Intell.*, 150:59–95, 2003.

[17] L. Karl Branting. *Reasoning with Rules and Precedents: A Computational Model of Legal Analysis*. Springer Netherlands, 2000.

[18] L.Karl Branting. Building explanations from rules and structured cases. *International Journal of Man-Machine Studies*, 34(6):797 – 837, 1991. ISSN 0020-7373. doi: https://doi.org/10.1016/0020-7373(91)90012-V. URL http://www.sciencedirect.com/science/article/pii/002073739190012V. Al and Legal Reasoning. Part 1.

[19] Gerhard Brewka, Thomas Eiter, and Mirosław Truszczyński. Answer Set Programming at a Glance. *Commun. ACM*, 54(12):92âĂŞ103, December 2011. ISSN 0001-0782. doi: 10.1145/2043174.2043195. URL https://doi.org/10.1145/2043174.2043195.

[20] Stefanie Brüninghaus. Daniel: Integrating case-based and rule-based reasoning in law. In *AAAI*, 1994.

[21] Luca Cervone, Monica Palmirani, and Tommaso Ognibene. Legal rules, text and ontologies over time. In *Proceedings of the RuleML2012@ECAI Challenge, at the 6th International Symposium on Rules*, volume 874, 01 2007.

[22] Tyson Condie, Ariyam Das, Matteo Interlandi, Alexander Shkapsky, Mohan Yang, and Carlo Zaniolo. Scaling-up reasoning and advanced analytics on big-

data. *Theory and Practice of Logic Programming*, 18(5-6):806âĂŞ845, 2018. doi: 10.1017/S1471068418000418.

[23] Ariyam Das, Sahil M. Gandhi, and Carlo Zaniolo. Astro: A datalog system for advanced stream reasoning. In Alfredo Cuzzocrea, James Allan, Norman W. Paton, Divesh Srivastava, Rakesh Agrawal, Andrei Z. Broder, Mohammed J. Zaki, K. SelÃğuk Candan, Alexandros Labrinidis, Assaf Schuster, and Haixun Wang, editors, *CIKM*, pages 1863–1866. ACM, 2018. ISBN 978-1-4503-6014-2. URL `http://dblp.uni-trier.de/db/conf/cikm/cikm2018.html#DasGZ18`.

[24] Cristhian A. D. Deagustini, S. E. F. Dalibón, Sebastian Gottifredi, Marcelo A. Falappa, C. Chesñevar, and Guillermo R. Simari. Defeasible argumentation over relational databases. *Argument Comput.*, 8:35–59, 2017.

[25] Mauro Dragoni, Serena Villata, Williams Rizzi, and Guido Governatori. Combining natural language processing approaches for rule extraction from legal documents. In Ugo Pagallo, Monica Palmirani, Pompeu Casanovas, Giovanni Sartor, and Serena Villata, editors, *AI Approaches to the Complexity of Legal Systems - AICOL International Workshops 2015-2017: AICOL-VI@JURIX 2015, AICOL-VII@EKAW 2016, AICOL-VIII@JURIX 2016, AICOL-IX@ICAIL 2017, and AICOL-X@JURIX 2017, Revised Selected Papers*, volume 10791 of *Lecture Notes in Computer Science*, pages 287–300. Springer, 2017.

[26] P. M. Dung. On the Acceptability of Arguments and Its Fundamental Role in Nonmonotonic Reasoning, Logic Programming, and n-Person Games. *Artificial Intelligence*, 77(2):321–357, 1995.

[27] Thomas Eiter and Rafael Kiesel. Weighted lars for quantitative stream reasoning. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, SenÃľn Barro, Alberto Bugarín, and Jérôme Lang, editors, *ECAI*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 729–736. IOS Press, 2020. ISBN 978-1-64368-101-6. URL `http://dblp.uni-trier.de/db/conf/ecai/ecai2020.html#EiterK20`.

[28] Gabriela Ferraro, Ho-Pun Lam, Silvano Colombo Tosatto, Francesco Olivieri, Mohammad Badiul Islam, Nick van Beest, and Guido Governatori. Automatic extraction of legal norms: Evaluation of natural language processing tools. In Maki Sakamoto, Naoaki Okazaki, Koji Mineshima, and Ken Satoh, editors, *JSAI-isAI International Workshops, JURISIN*, volume 12331 of *Lecture Notes in Computer Science*, pages 64–81. Springer,

2019. doi: 10.1007/978-3-030-58790-1_5. URL https://doi.org/10.1007/978-3-030-58790-1_5.

[29] Anne von der Lieth Gardner. *An Artificial Intelligence Approach to Legal Reasoning*. MIT Press, Cambridge, MA, USA, 1987. ISBN 0-262-07104-5.

[30] Marco Gavanelli, Evelina Lamma, Fabrizio Riguzzi, Elena Bellodi, Riccardo Zese, and Giuseppe Cota. Abductive logic programming for normative reasoning and ontologies. In *JSAI-isAI Workshops*, volume 10091 of *Lecture Notes in Computer Science*, pages 187–203, 2015.

[31] El Ghosh. *Automation of legal reasoning and decision based on ontologies*. PhD thesis, INSA de Rouen, 2018.

[32] T. F. Gordon, H. Prakken, and D. N. Walton. The Carneades model of argument and burden of proof. *Artificial Intelligence*, 171(10-15):875–896, 2007. ISSN 0004-3702.

[33] Guido Governatori. Representing business contracts in RuleML. *International Journal of Cooperative Information Systems*, 14(2-3):181–216, June-September 2005.

[34] Guido Governatori. On the relationship between Carneades and defeasible logic. In *Proceedings of the ICAIL 2011*, pages 31–40. ACM, 2011.

[35] Guido Governatori. The Regorous approach to process compliance. In *2015 IEEE 19th International Enterprise Distributed Object Computing Workshop*, pages 33–40. IEEE Press, 2015. doi: 10.1109/EDOC.2015.28.

[36] Guido Governatori and Mustafa Hashmi. No time for compliance. In *Enterprise Distributed Object Computing Conference (EDOC), 2015 IEEE 19th International*, pages 9–18. IEEE, 2015.

[37] Guido Governatori and Antonino Rotolo. Defeasible logic: Agency, intention and obligation. In *Proceedings of the DEON 2004*, number 3065 in LNCS, pages 114–128. Springer, 2004.

[38] Guido Governatori and Antonino Rotolo. Bio logical agents: Norms, beliefs, intentions in defeasible logic. *Autonomous Agents and Multi-Agent Systems*, 17 (1):36–69, 2008.

[39] Guido Governatori and Antonino Rotolo. A conceptually rich model of business process compliance. In *Proceedings of the APCCM 2010*, number 110 in CRPIT, pages 3–12. ACS, 2010.

[40] Guido Governatori and Sidney Shek. Regorous: A business process compliance checker. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law*, pages 245–246, 2013. doi: 10.1145/2514601.2514638.

[41] Guido Governatori, Francesco Olivieri, Antonino Rotolo, and Simone Scannapieco. Computing Strong and Weak Permissions in Defeasible Logic. *J. Philosophical Logic*, 42(6):799–829, 2013.

[42] Guido Governatori, Pompeu Casanovas, and Louis de Koker. On the formal representation of the australian spent conviction scheme. In Víctor Gutiérrez Basulto, TomÃąš Kliegr, Ahmet Soylu, Martin Giese, and Dumitru Roman, editors, *4th International Joint Conference on Rules and Reasoning (RuleML-RR 2020*, volume 12173 of *LNCS*, pages 177–185, Cham, 2020. Springer International Publishing. doi: 10.1007/978-3-030-57977-7_14.

[43] Risto Hilpinen. Deontic logic. In Lou Goble, editor, *The Blackwell Guide to Philosophical Logic*. Wiley-Blackwell, 2001.

[44] Jesper Hoeksema and Spyros Kotoulas. High-performance Distributed Stream Reasoning using S4. In *Proccedings of the 1st International Workshop on Ordering and Reasoning*, 2011.

[45] Rinke Hoekstra, Joost Breuker, Marcello Di Bello, Alexander Boer, et al. The LKIF Core Ontology of Basic Legal Concepts. *LOAIT*, 321:43–63, 2007.

[46] Anna Huggings, Alice Witt, Nicholas Suzor, Mark Burdon, and Guido Governatori. Financial technology and regulatory technology, issues paper submission. Technical Report Submission 196, Select Senate Committee on Financial Technology and Regulatory Technology, December 2020.

[47] Mohammad Badiul Islam and Guido Governatori. Ruleoms: A rule-based online management system. In Katie Atkinson, editor, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Law*, pages 187–191, New York, 2015. ACM.

[48] Mohammad Badiul Islam and Guido Governatori. RuleRS: a rule-based architecture for decision support systems. *Artificial Intelligence and Law*, 2018. ISSN 1572-8382.

[49] Robert Kowalski and Anthony Burton. WUENIC - A Case Study in Rule-Based Knowledge Representation and Reasoning. In Manabu Okumura, Daisuke Bekki, and Ken Satoh, editors, *JSAI-isAI Workshops*, volume 7258 of *Lecture*

*Notes in Computer Science*, pages 112–125. Springer, 2011. ISBN 978-3-642-32089-7.

[50] Brian Lam and Guido Governatori. Towards a model of UAVs navigation in urban canyon through defeasible logic. *Journal of Logic and Computation (JLC)*, 23(2):373–395, 2013.

[51] Ho-Pun Lam and Guido Governatori. The Making of SPINdle. In Guido Governatori, John Hall, and Adrian Paschke, editors, *RuleML*, volume 5858 of *Lecture Notes in Computer Science*, pages 315–322. Springer, 2009. ISBN 978-3-642-04984-2.

[52] Antoni Ligeza. *Logical Foundations for Rule-Based Systems, 2nd Ed.* Springer, 01 2006. ISBN 978-3-540-29117-6. doi: 10.1007/3-540-32446-1.

[53] Qing Liu, Mohammad Badiul Islam, and Guido Governatori. Towards an efficient rule-based framework for legal reasoning. *unders submission*, 2021.

[54] Vaughne Miller. How much legislation comes from Europe? House of Commons Library Research Paper, 10-62, 13 October 2010.

[55] Rohan Nanda, Luigi Di Caro, Guido Boella, Hristo Konstantinov, Tenyo Tyankov, Daniel Traykov, Hristo Hristov, Francesco Costamagna, Llio Humphreys, Livio Robaldo, and Michele Romano. A unifying similarity measure for automated identification of national implementations of european union directives. In Jeroen Keppens and Guido Governatori, editors, *Proceedings of the 16th edition of the International Conference on Artificial Intelligence and Law, ICAIL 2017, London, United Kingdom, June 12-16, 2017*, pages 149–158. ACM, 2017.

[56] M. Negnevitsky. *Artificial Intelligence: A Guide to Intelligent Systems*. Addison-Wesley, 2002. ISBN 9780201711592. URL `https://books.google.com.au/books?id=PgxmQgAACAAJ`.

[57] Monica Palmirani, Michele Martoni, Arianna Rossi, Cesare Bartolini, and Livio Robaldo. Pronto: Privacy ontology for legal compliance. In *Proceedings of the 18$^{th}$ European Conference on Digital Government (ECDG)*, October 2018. Forthcoming.

[58] Bijan Parsia, Evren Sirin, Bernardo Cuenca Grau, Edna Ruckhaus, and Daniel Hewlett. Cautiously Approaching SWRL. Preprint submitted to Elsevier Science, 2005.

[59] V. Paul and R. V. Polamraju. Rule management and inferencing in relational databases. In *IEEE Proceedings of the SOUTHEASTCON '91*, pages 695–697 vol.2, April 1991. doi: 10.1109/SECON.1991.147846.

[60] Henry Prakken. An Abstract Framework for Argumentation with Structured Arguments. *Argument and Computation*, 1(2):93–124, 2009.

[61] Henry Prakken and Giovanni Sartor. Law and logic: A review from an argumentation perspective. *Artif. Intell.*, 227:214–245, 2015.

[62] Jim Prentzas and Ioannis Hatzilygeroudis. Categorizing approaches combining rule-based and case-based reasoning. *Expert Systems*, 24:97–122, 2007.

[63] Tore Risch, René Reboh, Peter E. Hart, and Richard O. Duda. A functional approach to integrating database and expert systems. *Commun. ACM*, 31(12): 1424–1437, 1988.

[64] Edwina L. Rissland and David B. Skalak. Cabaret: Rule interpretation in a hybrid architecture. *International Journal of Man-Machine Studies*, 34:839–887, 1991.

[65] L. Robaldo, C. Bartolini, M. Palmirani, A. Rossi, M. Martoni, and G. Lenzini. Formalizing GDPR provisions in reified I/O logic: the DAPRECO knowledge base. *The Journal of Logic, Language, and Information*, In press., 2020.

[66] Livio Robaldo and Xin Sun. Reified Input/Output logic: Combining Input/Output logic and Reification to represent norms coming from existing legislation. *Journal of Logic and Computation*, 27(8):2471–2503, 04 2017. doi: 10.1093/logcom/exx009. URL `https://dx.doi.org/10.1093/logcom/exx009`.

[67] Ken Satoh, Kento Asai, Takamune Kogawa, Masahiro Kubota, Megumi Nakamura, Yoshiaki Nishigai, Kei Shirakawa, and Chiaki Takano. PROLEG: An Implementation of the Presupposed Ultimate Fact Theory of Japanese Civil Code by PROLOG Technology. In *JSAI-isAI Workshops*, volume 6797 of *Lecture Notes in Computer Science*, pages 153–164. Springer, 2010.

[68] Marek J. Sergot. A computational theory of normative positions. *ACM Trans. Comput. Log.*, 2(4):581–622, 2001.

[69] Marek J. Sergot, Fariba Sadri, Robert A. Kowalski, F. Kriwaczek, Peter Hammond, and H. T. Cory. The British Nationality Act as a Logic Program. *Commun. ACM*, 29(5):370–386, 1986.

[70] Mark Snaith and Chris Reed. TOAST: Online ASPIC+ implementation. In Bart Verheij, Stefan Szeider, and Stefan Woltran, editors, *Proc. of the 4th International Conference on Computational Models of Argument (COMMA 2012)*, volume 245 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2012.

[71] Michael Stonebraker. The integration of rule systems and database systems. *IEEE Trans. Knowl. Data Eng.*, 4:415–423, 1992. This paper provides a survey on rule and database integration for a decade. The author discuss possible issue with separating two systems as consistancy of the data and knowledge base is not guranteed. Instead of coupling two system it is better to intergrate two systems. The paper also discuss the classfication and implementation of DBMS rules system.

[72] Rudi Studer, V.Richard Benjamins, and Dieter Fensel. Knowledge engineering: Principles and methods. *Data and Knowledge Engineering*, 25 (1):161 – 197, 1998. ISSN 0169-023X. doi: https://doi.org/10.1016/S0169-023X(97)00056-6. URL `http://www.sciencedirect.com/science/article/pii/S0169023X97000566`.

[73] Jacopo Urbani, Alessandro Margara, Ceriel J. H. Jacobs, Frank van Harmelen, and Henri E. Bal. DynamiTE: Parallel Materialization of Dynamic RDF Data. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul T. Groth, Chris Biemann, Josiane Xavier Parreira, Lora Aroyo, Natasha F. Noy, Chris Welty, and Krzysztof Janowicz, editors, *The Semantic Web – ISWC 2013*, volume 8218 of *Lecture Notes in Computer Science*, pages 657–672. Springer, 2013. ISBN 978-3-642-41334-6.

[74] Saskia Van de Ven, Joost Breuker, Rinke Hoekstra, and Lars Wortel. Automated Legal Assessment in OWL 2. In Enrico Francesconi, Giovanni Sartor, and Daniela Tiscornia, editors, *JURIX*, volume 189 of *Frontiers in Artificial Intelligence and Applications*, pages 170–175. IOS Press, 2008. ISBN 978-1-58603-952-3.

[75] Saskia Van de Ven, Rinke Hoekstra, Joost Breuker, Lars Wortel, and Abdallah El-Ali. Judging Amy: Automated Legal Assessment using OWL 2. In Catherine Dolbear, Alan Ruttenberg, and Ulrike Sattler, editors, *OWLED*, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[76] Michelle J. White. Legal complexity and lawyers' benefit from litigation. *International Review of Law and Economics*, 12(3):381 – 395, 1992. ISSN

0144-8188. doi: https://doi.org/10.1016/0144-8188(92)90016-K. URL http://www.sciencedirect.com/science/article/pii/014481889290016K.

[77] Adam Z. Wyner and Wim Peters. On rule extraction from regulations. In Katie Atkinson, editor, *Legal Knowledge and Information Systems - JURIX 2011: The Twenty-Fourth Annual Conference, University of Vienna, Austria, 14th-16th December 2011*, volume 235 of *Frontiers in Artificial Intelligence and Applications*, pages 113–122. IOS Press, 2011. doi: 10.3233/978-1-60750-981-3-113. URL https://doi.org/10.3233/978-1-60750-981-3-113.

[78] Mohan Yang, Alexander Shkapsky, and Carlo Zaniolo. Scaling up the performance of more powerful datalog systems on multicore machines. *VLDB J.*, 26(2):229–248, 2017. URL http://dblp.uni-trier.de/db/journals/vldb/vldb26.html#YangSZ17.

 Received 1 February 2021