Journal of applied logics (Print) ISSN 2631-9810
Journal of applied logics (Online) ISSN 2631-9829

Journal of Applied Logics

The IfCoLog Journal of Logics and their Applications

Volume 8 • Issue 4 • May 2021

Available online at
www.collegepublications.co.uk/journals/ifcolog/
Free open access

Journal of Applied Logics - IfCoLog
Journal of Logics and their Applications

Volume 8, Number 4

May 2021

Disclaimer

Statements of fact and opinion in the articles in Journal of Applied Logics - IfCoLog Journal of Logics and their Applications (JALs-FLAP) are those of the respective authors and contributors and not of the JALs-FLAP. Neither College Publications nor the JALs-FLAP make any representation, express or implied, in respect of the accuracy of the material in this journal and cannot accept any legal responsibility or liability for any errors or omissions that may be made. The reader should make his/her own evaluation as to the appropriateness or otherwise of any experimental technique described.

ISBN 978-1-84890-366-1
ISSN (E) 2631-9829
ISSN (P) 2631-9810

College Publications
Scientific Director: Dov Gabbay
Managing Director: Jane Spurr

http://www.collegepublications.co.uk

Editorial Board

Scope and Submissions

This journal considers submission in all areas of pure and applied logic, including:

pure logical systems
proof theory
constructive logic
categorical logic
modal and temporal logic
model theory
recursion theory
type theory
nominal theory
nonclassical logics
nonmonotonic logic
numerical and uncertainty reasoning
logic and AI
foundations of logic programming
belief change/revision
systems of knowledge and belief
logics and semantics of programming
specification and verification
agent theory
databases
dynamic logic
quantum logic
algebraic logic
logic and cognition
probabilistic logic
logic and networks
neuro-logical systems
complexity
argumentation theory
logic and computation
logic and language
logic engineering
knowledge-based systems
automated reasoning
knowledge representation
logic in hardware and VLSI
natural language
concurrent computation
planning

This journal will also consider papers on the application of logic in other subject areas: philosophy, cognitive science, physics etc. provided they have some formal content.

Submissions should be sent to Jane Spurr (jane@janespurr.net) as a pdf file, preferably compiled in LaTeX using the IFCoLog class file.

Contents

ARTICLES

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.

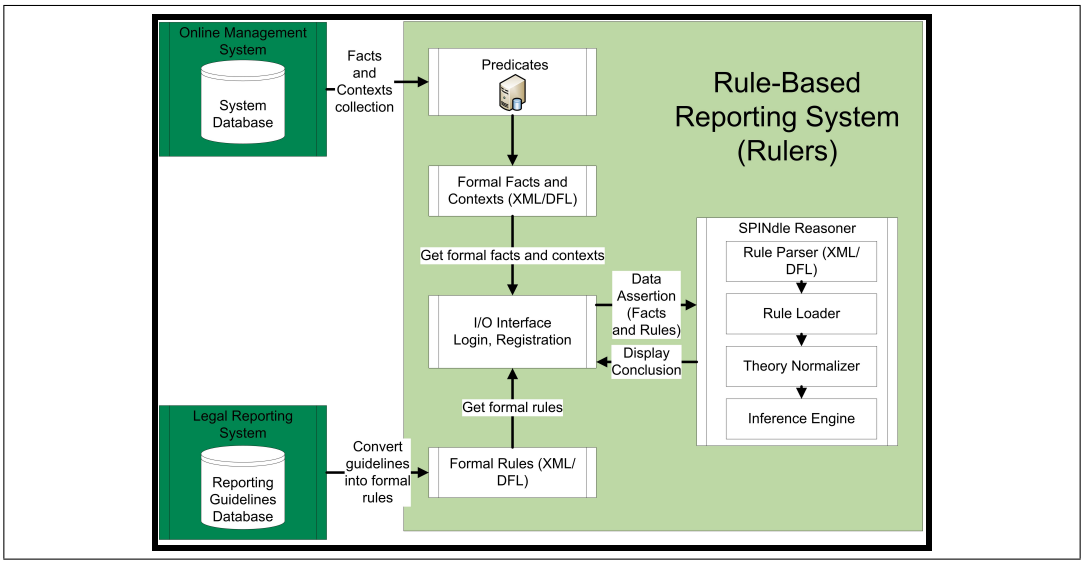


Figure 1: Rule-based Reporting System (RuleRS)

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 18th 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

NLP techniques for normative mining

Gabriela Ferraro
Data61, CSIRO, Black Mountain, Australia
gabriela.ferraro@data61.csiro.au

Ho-Pun Lam
Data61, CSIRO, Eveleigh, Australia
brian.lam@data61.csiro.au

Abstract

Natural Language Processing (NLP) is a branch of artificial intelligence that study the interactions between computers and human (natural) language. In the field of legal informatics, the focus has been centered on mining and formalising normative information such that the legal norms extracted can be interpreted and reasoned by machines in an automated fashion. In this article, we focus our attention on discussing the challenges of normative mining from a NLP perspective, and present a detailed overview of existing techniques on semantic parsing, and their strengths and limitations on mining legal norms.

1 Introduction

Natural Language Processing (NLP) is a branch of artificial intelligence that study the interaction between computers and human (natural) language. NLP deals with the design, development and analysis of computational algorithms for processing natural language, and its theoretical foundations encompasses computer science, mathematics, and linguistics. One of goals of NLP is to provide language technology capabilities, for example, translating between languages, extracting information from texts, answering questions, holding a conversation, taking instructions, and so on.

Modern methods rely mostly on data-driven machine learning algorithms that learn from examples. Other important methods for language processing are deterministic models (e.g., state machines), declarative models (e.g., rule systems), logic (e.g., predicate calculus), and probabilistic methods (e.g., weighted state machines).

Some of the characteristics of human language makes NLP particularly challenging, for instance, human language data is discrete, thus it is not possible to

gradually approach a solution; language follows a power law distribution [85], which means that algorithms have to deal with observations never seeing before; language is compositional, for example, words combine to create larger units as phrases, hence algorithms have to model implicit recursivity; and language is ambiguous, thus a construction may have multiple interpretations [40]. Despite the difficulties of modelling language, recent NLP methods has demonstrated impressive improvements in complex task as machine translation [74] and question answering [73].

Advances on NLP research are central for modelling the problem of formalising normative information from legal documents. The ultimate goal of normative mining is to automate the extraction and formalisation of laws and regulations in a format that a machine can interpret and reason about. This is a complex and challenging task as it encompasses language understanding at different processing levels: document-collection level, document level, section level, and sentence level. Sentence level processing is especially hard since it require semantic analysis of complex utterances, for example, identification of predicates and their arguments in particularly long sentences. Indeed, the process of extracting normative information from sentences is so complex that, at the moment, it is mostly done manually.

This chapter is intended to provide a review and discussion of the problems and plausible approaches in automating normative mining from legal documents with an NLP perspective. The rest of the chapter is organised as follows. Section 2 discusses the issues and problems of automatically extract the normative information from legal documents. Section 3 discuss the possible approaches to normative mining, and a literature review of the state-of-the-art approaches will be presented herein. Problems related to evaluation methodologies and benchmarking for normative mining will be discussed in Section 4. Section 5 presents preliminary results to normative rules extraction using relation extraction and semantic parsing models. Finally, Section 6 present the conclusions and some of the directions for future research in this area.

2 What are the Problems?

Algorithms developed by the NLP community has been used to model the process of extracting normative rules from legal documents with certain success. Extracting normative rules is difficult because it requires sentence semantic analysis, which is one of the most challenging NLP problems. Moreover, some characteristics of legal documents also possess additional problems for NLP methods. Legal documents considerable differ from other documents such as news or Wikipedia articles that are usually used in NLP research. Those differences make off-the-shelf NLP solu-

tions sometimes hard to apply, or insufficient, or inefficient, when analysing legal documents. In particular, legal documents have a peculiar document structure that needs to be captured in order to properly interpret its written content. Within the documents, sentences can be extremely long and complex, which possess important challenges to the semantic analysis methods available. In this section, we review some of the NLP-related challenges towards the formalisation of legal norms. In particular, we elucidate on the following.

- Document level cross-referencing
- Ambiguity and inconsistent terminology
- Sentence semantic analysis
- Identification of normative effects

While dealing with ambiguity, inconsistent terminology and sentence semantic analysis are central problems in NLP, handling sentence complexity and cross-referencing are particular to the legal domain. In what follows, each challenge will be described in detail so that the landscape of the problem and situations that led to the problem can be explored.

2.1 Document Level Cross-referencing

Typically, a legal document is structured into different chapters, articles, sections and subsections, where each of these might contain one or several sentences or even paragraphs. Consider the example as shown in Figure 1 illustrating the structure of the proposed EU regulation on EU administrative law [25]. As can be seen from the figure, the proposed regulation structure is constituted by four main chapters with thirty different articles, which correspond to different stages of the procedure, namely: (i) *initiation* of the procedure, (ii) *management* of the procedure, (iii) *conclusion* of the procedure, (iv) rights related to *rectification* and *withdrawal* of the act. In addition to this, it also consists of a chapter on general provisions related to the *objective*, *scope* and *definitions* (such as terms and concepts being used) of the act; and another chapter on the general scope, such as *evaluation*, and when the act enter into force, etc.

Essentially, each statement in a legal document has their own specific *goals*, *objectives*, and *scopes*, which define the *context* under which a particular set of statements become applicable and the (normative) *effects* that follow from applying it. This modularity nature of legislation allows legal drafters to focus on a particular aspect of legislation when drafting the document. Hence, referencing information from one section to another within the same regulation, or to other regulations, is not uncommon in legal documents.

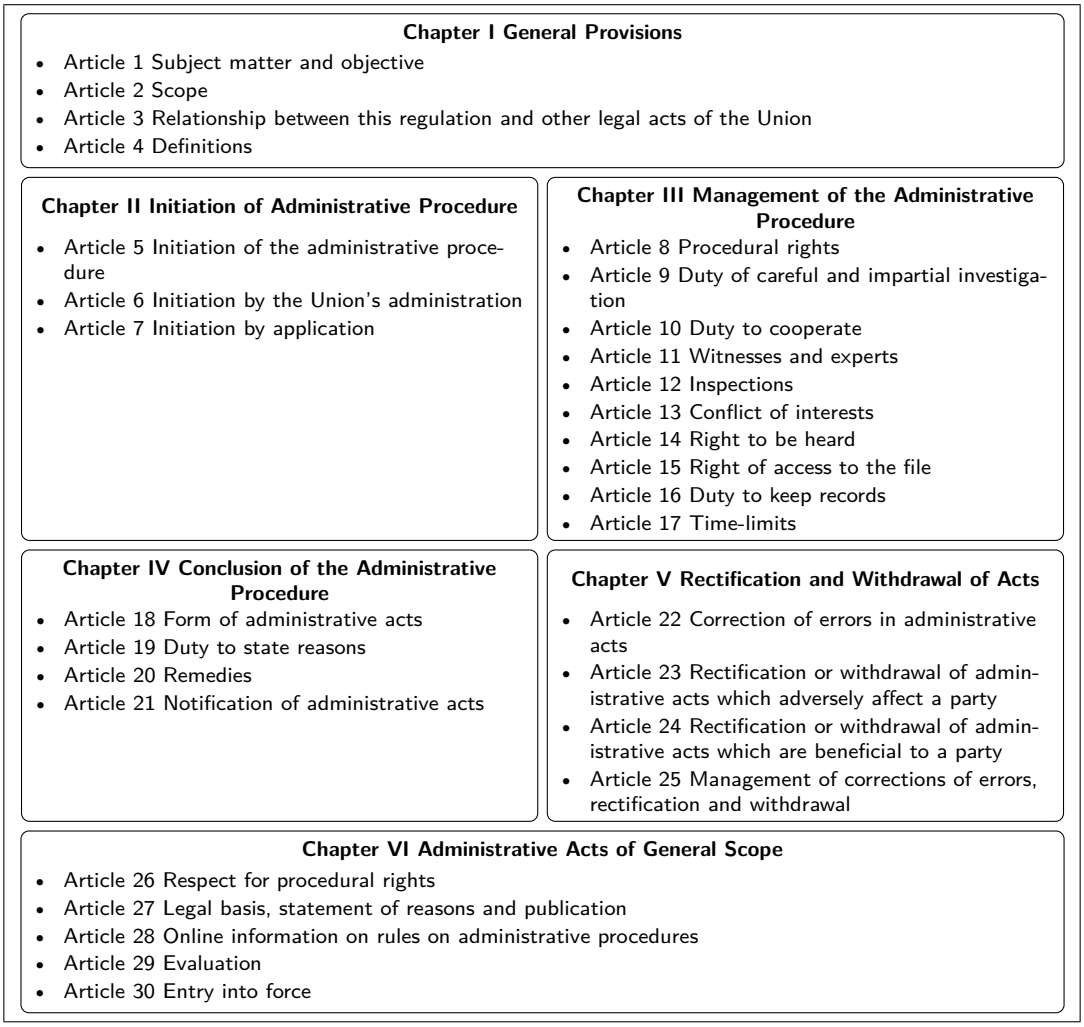


Figure 1: Structure of the proposed draft regulation from European Parliamentary Research Service (adopted from [25]).

Technically, *cross-references* are explicit phrases that appear in regulations and can be used to link regulatory requirements within and across regulations [32]. They can help to avoid ambiguity that may occur across different sections of the documents and can help to indicate whether a sentence is an *elaboration*, *subordinate*, or *prevailing* with respect to other sentences or definitions. They can also be used to conferring a priority to reconcile potential conflicts by discarding existing goals or substituting alternative top-level goals [55]. Hence, from NLP point of view,

1.2 Natural ventilation – General

⋮

1.2.2 Natural ventilation of occupied spaces must be achieved by providing a net openable area of windows or other openings to the outside of no less than 5% of the floor area. The 5% floor area requirement does not apply to:

a) occupied spaces in Commercial **and** Industrial buildings where products listed in NZBC Clause G4.3.3 are generated (mechanical ventilation of these spaces is required), **and**

b) household units **and** accommodation units where there is only one external wall with opening windows (refer to Paragraph 1.3 for additional requirements if natural ventilation is used).

Figure 2: Example of logical ambiguity (Sentence extracted from [56]).

identifying cross-references is an important task in normative mining as they define the context of linguistic utterances and can help to resolve referential and lexical ambiguity, which will be discussed below.

2.2 Ambiguity and Inconsistent Terminology

As in other language related tasks, ambiguity is one of the problems to deal with in normative mining. We often encounter *lexical ambiguity* due to polysemy, *syntactic (or structural) ambiguity* due to different interpretations of a sentence grammatical structure, *referential ambiguity* that happens when it is not clear to what or whom a concept refers, and *logical ambiguity*, which leads to different logic interpretations.

Drafters of legal documents try to avoid ambiguity and ideally, produce a document that results in only one interpretation (e.g., avoid pronouns, avoid synonyms to refer to the same concept, add attributes to identify parties, use punctuation to define the scope of quantifiers, etc.). Furthermore, to avoid lexical ambiguity, legal documents usually include a glossary, sometimes named as Definitions (see Figure 1), which list the most important lexical items from the document and their corresponding definitions. For example, a regulation about *buildings* may contained a glossary with entries such as *commercial building*, *industrial building*, *accommodation units*, etc. However, as natural language is used to write the legislation, unintended ambiguities may arise. Lexical and referential ambiguities are usually inferred from the context in which the lexical units appeared. In the case of processing legal documents, in addition to the current (local) context, conditions related to the meaning of linguistic lexical units sometimes need to be inherited from other document sections via cross-referencing (see Section 2.1).

Logical ambiguity, on the other hand, refers to the use of natural language that

can be mapped to different logical interpretations [13]. Consider the fragment of legislation extracted from [56], as shown in Figure 2. Syntactically the terms *commercial building* and *industrial building* in the first sub-condition (a), *household units* and *accommodation units* in the second sub-condition (b), and the two sub-conditions are connected using *and*. However, logically (or semantically), the statement is in fact representing conditions to the four different types of building and should be represented using disjunction, i.e., *or*, in the resulting normative rule(s) generated.

Apart form the ambiguity problems just mentioned, the inconsistent use of terminology across different documents may also affect the normative mining process as the same entity can be expressed using different terms or the scope of the same term has been defined differently. As mentioned in [4],

> *[. . .] inconsistencies within a single directive could have significant consequences for the national implementation law, instigating legal uncertainty in commercial and legal practices.*

Consider the case in EU as an example. Due to the divergence of languages being used, discrepancies among language versions are capable of aggravating the adverse consequence and may cause the same regulatory inconsistencies [4, §2.4]. Hence, the European Commission has declared a requirement to the EU legislature to eliminate differences in terms and concepts that cannot be explained by differences in the problems being addressed [4, §2.4.2.1]. Recently, [6] studied the role of EU legal English as a *lingua franca* in shaping EU legal culture; while [58] studied the terminological inconsistencies problem arising from *term-aliasing* (of the same language), i.e., when the same entity has been referred to by multiple terms across a corpus of documents related to a particular context, so that a better refinement, and possibly unified, glossary of terms can be used across the set of documents.

However, as the problem of dealing with inconsistent terminology does not directly affect the normative rules generation process, it will be skipped for the rest of the chapter.

2.3 Sentence Semantic Analysis

The automatic extraction of normative rules from sentences written in natural language required some sort of sentence semantic analysis that produce a meaning representation, for example, a logical statement or a deterministic rule, that is executable by a machine. An expressive meaning representation involves many aspects such as meaning of words, meaning associated with grammatical constructions (syntax), knowledge about the discourse structure (relationship between phrases and

Sentence	*Accessible showers **shall have** a level threshold.*
Normative rule	$accessibleShowers(X) \Rightarrow_{\mathrm{O}} haveLevelThreshold(X)$

Table 1: Example of Deontic Modalities identification and attachment (*obligation* (O))

sentences), common-sense knowledge about the topic of the sentence, and pragmatic knowledge about the context where the discourse is occurring [40].

The first problem encountered here is that, currently, there is no NLP computational framework that encompasses all the above aspects of meaning representation. The state-of-the-art semantic analysis methods produce meaning representations by analysing the lexical and grammatical (syntactical) characteristics of language at the sentence level [61, 79, 83, 84, 49, 68] and should be considered fairly modest in their scope.

The second problem is that the state-of-the-art semantic analysis struggled to capture predicates in complex sentences. Capturing sentence predicates units such as verbs and their arguments is essential to model meaning. This is particularly challenging in legal domain because sentences from legal documents can be extremely long and contain multiple predicates. As stated by [15], sentence length is an indication of sentence complexity. While the average number of lexical units in a sentence written in the English Wikipedia is about nineteen [81], sentences from legal documents can have more than fifty units, as can be seen in Figure 2. Long sentences tend to have a complex grammatical structure, usually contained several predicates and coordinate and subordinate constructions, and are likely to be poorly analysed even with the use of state-of-the-art syntax analysers. For example, it is well known that automatic methods for syntax analysis straggled to capture the scope of multiple coordinate conjunctions and antecedent of subordinate phrases [15]. Similarly, they also straggled to capture long distance dependencies, which sometimes are essential for capturing the scope of predicate arguments.

2.4 Identification of Normative Effects

Capturing the normative effects of legal norms is a crucial task in legal informatics. Typically, a regulation can be seen as a set of *provisions*, carried by speech acts, where a provision can assume different types as *definition*, *obligation*, *sanction*, *competence*, *amendments*, etc. [9], that are written in a highly structured way and in *legalese*. For instance, consider the fragment of regulation extracted from [57], as shown in Table 1. While it can be consider as a part of the physical structure of a

legislative text [28], it can also be qualified as a provision of type *obligation*, whose arguments are:

Subject:	Accessible showers
Predicate:	shall have
Object:	level threshold

As can be seen, such information affect the way that we interpret the legislation from a semantic point of view and the (normative) effects that specify the types of behaviour that it generated or permitted from applying those norms. Hence, it is foreseeable that detrimental impact (such as incorrect rights, duties, or obligations) or damaging effects may be conferred to some stakeholders if information like this is misinterpreted or has not been detected correctly.

3 Literature Review

In this section, we examine some promising NLP avenues related to the problem of capturing normative information from legal documents and legal statements, and review existing approaches found in literature. We consider three related work topics. In Section 3.1, we investigate different NLP approaches to capture sentences semantics and discuss their strengths and weaknesses. Techniques on identifying and resolving cross-referencing that appear among different lexical units are discussed in Section 3.2. Section 3.3 discusses the problem of capturing normative information from legal texts using NLP. Finally, Section 3.4 review existing approaches to deal with the problem of extracting legal norms from legal documents.

3.1 Capturing Sentence Semantics

Traditionally, NLP methods capture semantics through syntax. These approaches are based on the principle of compositionality [61], which follows the idea that the meaning of a sentence can be constructed by the meaning of its parts, and on the ordering and grouping of words and relations among words [37]. Recent data-driven methods learn the mapping from sentences to meaning representations without using syntax or other external knowledge. In what follows, we review some of the research streams to address the problem of producing some sort of meaning representation: syntax-driven semantic parsing, and neuronal generative models for semantic parsing

3.1.1 Syntax-Driven Semantic Parsing

The input of syntax-driven semantic parsers is a syntactic tree and the output is a meaning representation, e.g., a first order logic formula or a subject-verb-object

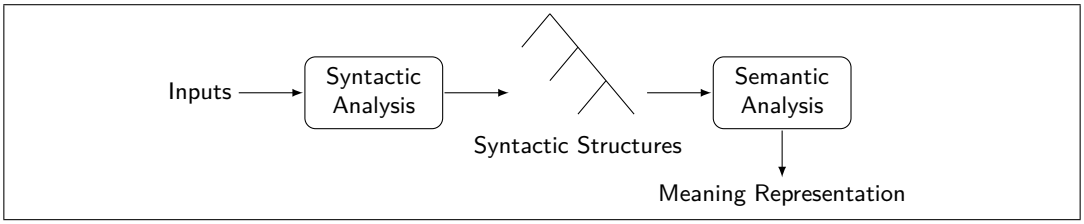


Figure 3: A simple pipeline approach to semantic analysis (adopted from [40, p. 596]).

triplet [40]. Figure 3 depicts a simple NLP pipeline approach to semantic analysis, in which an input is first passed through a parser to derive its syntactic structure, i.e., a syntax tree. The structure is then passed as input to a semantic analyser to produce a meaning representation.

In the literature, there are at least, the following NLP research streams that uses syntax as the basics for semantic analysis:

- Relation extraction
- Semantic role labelling
- Grammar-driven semantic parsing
- Data-driven semantic parsing

In what follows, we describe these research areas in more detail.

Relation Extraction refers to the extraction of relation tuples (typically binary relations) from plain text. For instance, from the sentence "*Buildings require ventilation*", it is possible to extract the tuple as shown below:

$$(require_{pred} = buildings_{arg1}, ventilation_{arg2})$$

where *pred* stands for *predicate* and arg_i denotes the *arguments of the predicate* and i indicates the *argument order* in its syntactic structure. Recent approaches to relation extraction are open to any relation, thus relations do not have to be defined in advance and are able to model long-range dependencies, making it possible to identify multiple relations inside a single sentence. Relation extraction can be also seen as a step towards the formulation of normative rules. After the predicate structure of a sentence is extracted, those building blocks still need to be instantiated in a rule format. However, a drawback of relation extraction is that it does not capture predicate nouns and adjectives, e.g., given the sentence *A large building is any building with a net lettable*

Figure 4: Example of Semantic Role Labelling for the sentence: *For the purposes of subclause 2.4, a person is responsible for an individual if the person is a parent of the individual.*

area greater than 300 m^2, the relation extraction tool OpenIE [2] extracted the following relation tuples: '*A large building*', '*is building with*', '*net lettable area greater than 300 m^2*', but the arguments of the predicate '*greater than*' are not extracted. A detailed review of the current state-of-the-art on relation extraction can be found in [66]

Semantic Role Labelling refers to the process that assigns labels to the semantic arguments or roles of predicates in a sentence such as *agent*, *goal*, *result*, among other semantic roles. This is an important step towards making sense of the meaning of a sentence. A semantic analysis of this sort is at a lower-level of abstraction than a syntax tree, e.g., it has more categories, thus groups fewer clauses in each category. It can be seen as an important step towards the extraction of normative rule since the main predicates and their arguments are the basic building blocks of a rule. For example, the sentence "*For the purposes of subclause 2.4, a person is responsible for an individual if the person is a parent of the individual*" would need four labels such as *verb*, *purpose*, *argument(s)*, and *adverbial*, as elicited in Figure 4[1]. The current state-of-the-art on semantic role labeling are deep learning models that uses syntax as [34].

Grammar-driven Semantic Parsing takes a sentence as input and output a meaning representation. The meaning representations can be diverse, e.g., a logical formula, a SQL query, a computer command, among others, as depicted in Table 2. Traditional approaches relied on categorical grammars, which are induced from data [84, 46, 45]. Another research stream has focused on the task of automatically learning the mapping from sentences to meaning representations based on a training set of sentences labelled with their semantics [30, 42, 52, 21, 68, 24]. Most of these works rely on manually designed syntactic and lexical features, which are usually domain and meaning representation dependent.

[1] The output of semantic role labeling in Figure 4 is generated using the tool: AllenNLP (`https://demo.allennlp.org/semantic-role-labeling`) (last accessed: 16 July 2019).

3.1.2 Neuronal Generative Models for Semantic Parsing

Inspired by recent research in machine translation [41, 74], neuronal generative models have been successfully applied to semantic parsing [21, 38, 50, 47]. These models are learn from sentences paired with meaning representations and do not use explicit syntactic knowledge. Indeed, the sequence-to-sequence models are able to generate translations directly from the probability distribution of the network without any external knowledge [41]. Thus, they do not rely on domain-dependent features and representation-specific solutions. One of the drawbacks is that these methods are data hungry and creating datasets for semantic parsing is not a trivial task.

The above methods captures semantic aspects from sentences, but are not design to capture normative rules explicitly. Even thought, they can provide certain level of automation to accomplish that task. Recent methods developed by the semantic parsing community are especially promising for facilitating the automation of normative rules extraction. However, they require the creation and release of datasets of sentences pairs with their corresponding normative rules.

3.2 Capturing Cross-referencing/Scope

Detection of document structure is one of the tools that can be used to determine the context of an argument in which it becomes effective or applicable, and can be used to enhance the quality of cross-referencing information that appear within the same document or among different legislation.

Input	Output
	Computer command
if I post something on blogger it will post it to wordpress	Blogger: Any new post $\Longrightarrow$ WordPress: Create a pos
	SQL
How many engine types did Val Musetti use?	SELECT COUNT Engine WHERE Driver = Val Musetti
	Lambda calculus
flight from Los Angeles to Phoenix	(lambda \$0 e (and (flight \$0) (from \$0 Los Angeles) (to \$0 Phoenix)))

Table 2: Examples of instances from three semantic parsing data set. The Input column refers to sentences written in natural language and the Output column refers to their corresponding mapping into different meaning representations: computer command, SQL, and Lambda calculus.

As mentioned before, legislations are presented following a very strict (and formal) structure. Based on this, [20] has categorized the structure of references into four different types, namely: (i) *simple references*, references to other legislations using their name, such as "Building Act", "part 1", or "Paragraph 1.3" (as shown in Figure 2); (ii) *complex references* or *multi-valued references*, a reference label that is constituted by more than one simple references, such as "article 1, section 2, paragraph 2"; (iii) *special cases*, references that contain a list or an exception. For example, a reference is made to the first item of article 1, the first item of article 2 and the first item of article 4 can can shorten to "the first item of articles 1, 2 and 4"; and (iv) *complete and incomplete references*, references that include complete (or respectively, incomplete) information of a particular document. For example, "section 1, article 1" is an incomplete references; while "section 1, article 1, Building Act" is a complete reference.

In addition to this, based on sequence of references appear, [75] has characterized references using the notions: *mention* to denote references that contain referring texts, and *antecedent* to denote the text that mentions refer to, and discussed the ways that it used in Japanese legislations.

In the past, regular expressions or context free grammar based approaches have been proposed to address this issue [20, 64, 75] and promising results have been reported in the literature. However, such approaches have been limited by the list of terms and abbreviations used, and are language dependent.

Recent trends in this area have been emphasised on the use of syntactic structures of the legal texts. For instance, [71] studied the problem of extracting and analysing semantic legal metadata (such as *actor*, *artifact*, *situation*, *action*, *constraint*, among others) using NLP-based extraction rules and proposed a conceptual model to capture the extracted information.

Falessi *et al.* [26], on the other hand, conducted an empirical evaluation on applying different NLP techniques (such as algebraic model (vector space model, latent semantic analysis, etc), term extraction, weighting schema, and similarity metric) and models to identify equivalent requirements (or linkage) across different documents, and concluded that simple measures are more precise than complex ones.

3.3 Capturing Normative Effects and Modalities

The conventions of legalese are not always use consistently and may introduce intended and unintended ambiguities that affect the performance of the automated tools [44]. To minimize the effects of such ambiguities, normative sentences are typically classified with the aid of some legal ontologies which explicitly specify a conceptualization, i.e., a formal description of concepts (i.e., the set of normative

effects in our case) either as a whole or focus on a particular aspect or activity, and their relations. It can be used to resolve the problem of language heterogeneities (and ambiguities) that appear in legal documents, and allow the information interchange between different diverse information systems [31].

Based on [39], [80] proposed an approach to extract legal concepts by parsing and extracting noun phrase in every sentence individually. [48] proposed an ontological approach to categorize and extract normative requirements from regulations, which helped knowledge engineers in rigorously identifying inconsistencies between the model and regulation. [14, 12] proposed an upper ontology, which has *two* tiers, for formalizing *frames* in legal provision. Deontic Concepts such as *permission*, *obligations*, *refrainments*, *exclusions*, *facts* and *definitions* are defined in the first tier; while the second tier describes concepts related to constituent phrases, such as *subject*, *acts*, *objects*, *purposes*, *instruments* and *locations*. [71] used syntactic structures and hand-written rules to identify entities (*agent*, *action*, *condition*, *exception*, among others) and linked action to a (normative/deontic) *modality* such as *obligations*, *permissions*, and *prohibitions*, and further divided constraints into *conditions*, *violations*, and *exceptions*, which are relevant in legal knowledge representation.

In [78], the authors have compared legal ontologies in three different dimensions, namely: (i) *epistemological adequacy* (such as *epistemological clarity*, *epistemological intuitiveness*, *epistemological relevance*, *epistemological completeness*, and *discriminative power*), (ii) *operationality* (such as *encoding bias*, *coherence*, and *computationality*), and (iii) *reusability* (such as *task-and-method reusability* and *domain reusability*). Their results have found that different ontology authors were using different conceptualisations of the legal domain despite their purposes were similar, and have not much agreement to what are the most important block of legal knowledge. However, their results also showed that there are not much differences in knowledge types distinguished but difference in priorities of these knowledge can be found. Recently, several legal ontologies have been proposed [8, 1, 69, 36], providing a more update-to-date conceptual architecture for the development of a legal system.

3.4 Related work

Algorithms developed by the NLP community has been used to model the process of extracting normative information from legal texts with certain success. [76] proposed an automated concept and norm extraction framework by exploiting the use of a Juridical (Natural) Language Constructs (JLC) as an intermediate format between the legal texts and a formal model. In their approach, the JLC is essentially a set of patterns that can appear in the legal document. Legal knowledge is identified and

constructed using noun and verb phrases patterns, that will later be transformed into formal rules. However, the effectiveness or efficiencies of their approach is still an unknown as no evaluation results has been reported in the paper.

Likewise, similar approaches have been used to transform legal text into an intermediate formalism, to improve the efficiencies of the legal norms generation process. For instance, [70, 7] used Semantics of Business Vocabulary and Rules (SBVR) [63] — a controlled natural language, as a tool to aid the information extraction process by clarifying ambiguities and inconsistencies that may appear in the regulations. [43, 62] proposed to convert sentences into a logical formalism that conform to Davidsonian Style [86], which is suitable for some languages that allow zero-pronouns (such as Japanese). In [77], the authors proposed an approach to translate legal texts into a formal language, such as UML, to support automatic legislation modelling. However, information about how much their system can help in reducing the burden of the normative information extraction process is still an unknown issue.

Instead of using pattern matching methods based on lexico-syntactic patterns, which are manually crafted or deduced automatically [3], [10] presented a technique to automatically extract semantic knowledge from legal texts based on the syntactic dependencies between terms extracted with a syntactic parser, which is also the technique used in [23] but with different kinds of information extracted. [82] proposed a linguistic oriented rule-based approach to extract deontic rule from regulations that build on top of General Architecture for Text Engineering (GATE) framework [18] and found that serious issues may appear when mapping thematic roles to syntactic position. [29], on the other hand, developed a translation systems called NL2KR to learn the semantics of unknown words from syntactically similar words with known meaning, which can later be translated into a variety of formal language representations. However, their evaluation was based on a few small sentences picked from the literature, and further enhancements is needed when dealing with long and more complicated sentences that frequently appear in legal documents.

The automated processing of extracting a certain types of information (such as the definitions of terms, metadata of a particular version of the regulation, etc.) is another challenging task that needs to be addressed. In the past, researchers in knowledge representation community has proposed different schemas, such as $^{\text{META}}$Lex [11], to encode legislation documents and content with annotated metadata in a structural and standardised format (mainly XML), which helps to improve the efficiency of managing and processing information in legal knowledge base systems. Maxwell *et al.* [54], on the other hand, proposed a legal taxonomy to address the problems, such as conflicting requirements, conflicting definition, refining an existing requirement, etc., that can appear when cross-referencing legal documents.

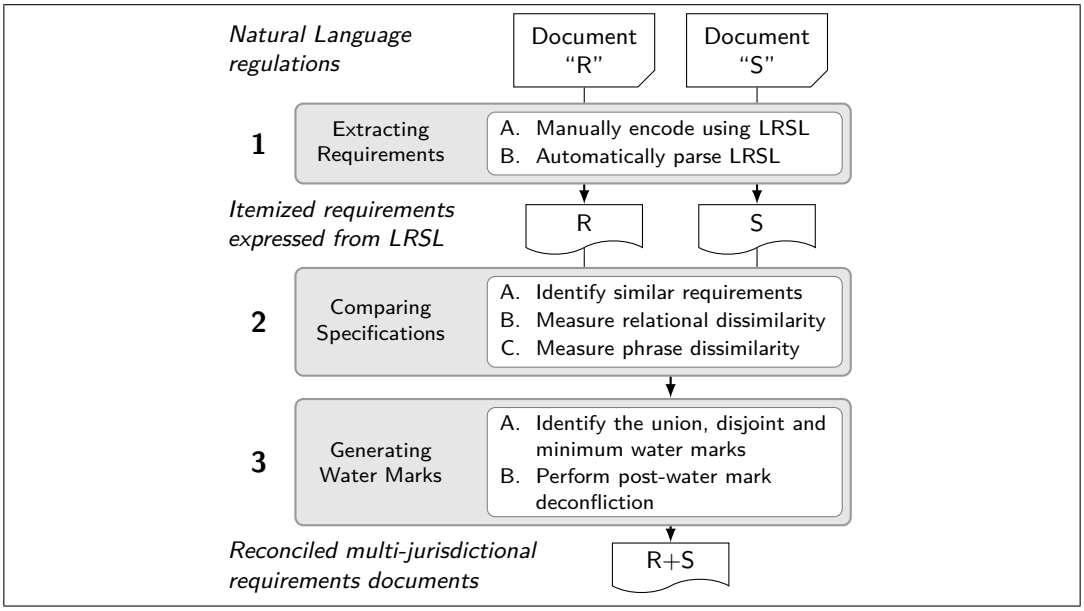


Figure 5: Overview of regulatory water mark construction (adopted from [32])

Gordon and Breaux *et al.* [32] developed a legal requirements specification language (LRSL) as a standard notation of extracted requirements and proposed a frame-based approach called "*requirements water marking*" to systematically align, manually extract and reconcile cross-references from multi-jurisdictions, as illustrated in Figure 5. Recently, LegalDocumentML [65], an OASIS standard, has been proposed to provide a common legal document standard to exchange legislative and judicial information between different institutions.

[9] proposed an automated framework for the semantic annotation of provisions to ease the retrieval process of norms, from the semantic, [19] presented a tool for extracting requirements from regulations where texts are annotated to identify fragments describing normative concepts, [44] present a tool for extracting requirements from regulations where texts are annotated to identify fragments describing normative concepts, and then a semantic model is constructed from these annotations and transformed into a set of requirements. [71] used the syntactic structures and hand-written rules to identify different types of entities (*agent*, *action*, *condition*, *exception*, among others), which are relevant in legal knowledge representation and can be useful in assisting the legal norms extraction process.

In another line of research, argumentation mining (AM) is intended to extract arguments from generic text corpora automatically, to provide structured data for

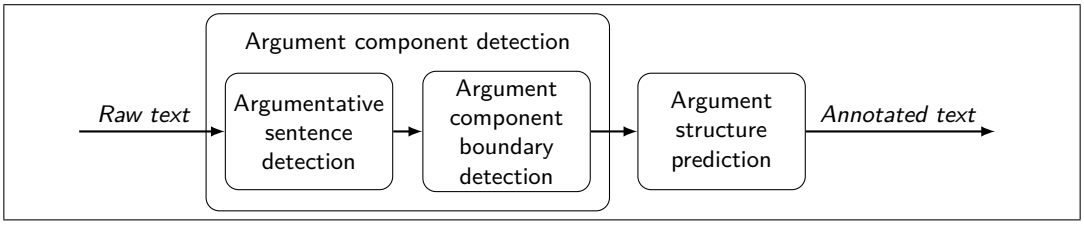


Figure 6: Argumentation Mining Pipeline architecture (adopted from [51])

computational models of argument and reasoning engines [17, 51, 60]. Most of the AM systems developed so far implement a pipeline architecture, as depicted in Figure 6, where a set of unstructured text are taken input and generate a set of structured/annotation document as output, and arguments and their relations are extracted and annotated so as to form an *argument graph* showing their linguistic structure, the relationship between different arguments, and recognizing the underlying concepts, evidences and consequences (a.k.a. conclusions or claims in some literature) of the legal texts.

Over the years, different techniques have been proposed to address a variety of tasks in the AM process. Most relevant to us, among others, are approaches that focus on the identification of argumentation structure, such as: *component identification*, *component classification*, and *structure identification* [72]. For instance, [60], classify arguments as either premises, claims, or proposes using convolution kernel methods with domain-dependent key phrases and text statistics, and have achieved an accuracy of 65%, which is a bit lower than [59] (73%). [16], on the other hand, adopt the notion of *textual entailment* to recognize features characterizing legal arguments and have applied different abstract argumentation frameworks over the set of generated arguments to determine the set of acceptable arguments with respect to the chosen semantics.

Legal documents considerably differ from other types of documents with respect to structure, language, length, etc. As a result, the desired generation of formalised legal norms may pose some significant challenges to the NLP community. In this section, we have provided an overview of the NLP approaches that have been developed for mining normative information from legal texts. Some research has been devoted to the task of generating legal norms from sentences in legal documents with some success [22] but are still not yet sufficient to cater the needs in practical use. Hence, more works need to be done to further enhance the efficiency and quality of normative information extracted. In the following section, we are going to address problems that may appear from evaluation and benchmarking perspectives.

4 Evaluation and Benchmarking

Reporting meaningful evaluations is essential for comparing, replicating and benchmarking methods, and is crucial for scientific progress and technical innovation [67]. However, evaluation and benchmarking have always been problematic in normative mining related research due to the lack of evaluation datasets, i.e., there are no available gold-standard with text snippets written in natural language paired with their corresponding normative rules, nor standards/tools to benchmark approaches that have been developed. Thus, making it difficult to compare and reproduce existing results.

In this section, we explore two fundamental questions concerning the evaluation of normative rule extraction: *what* to evaluate, and *how* to evaluate? We will response to each of these questions in turn and will discuss important aspects to be taken into account to prompt method comparisons and replications.

4.1 What to Evaluate?

A fundamental question in evaluation methodology is what to evaluate. On one hand, it is possible to *intrinsically* or directly evaluate a system on a set of desire functionalities, for example, the measure to what extent the automatically generated normative rules corresponds to correct outputs. On the other hand, it is possible to *extrinsically* assess the impact of a task external to the system, for example, the disclosure of reasoning of the generated normative rules.

NLP system are intrinsically evaluated in terms of *one input, one output* or *multiple outputs per input*. In the case of normative rules extraction, intrinsic evaluation should be done considering *one input, one output*, therefore avoiding multiple competing rules that can negatively affect the reasoning process, for example, by increasing complexity due to the number of rules.

Besides, it is also important to take into consideration the architecture of the proposed systems. Usually, NLP systems are not monolithic, but rather a set of modules in a processing pipeline. An advantage of implementing a system like this is that one can artificially manipulate a component input and observe its impact to the system's final effectiveness, and evaluation can be done at both an *individual* component level or at group level. For instance, syntax parsing heavily depends on the quality of the POS-Tags associated with each word. Artificially corrupting the POS-Tags can help to study the sensibility of the syntax parser to POS-Tagging errors. This is particularly relevant when evaluating modular architectures for normative rules extraction since the quality of off-the-shelf tools, such as POS-Taggers and syntax parsers, are seriously affected when analysing complicated sentences from

legal documents. Interestingly, modern NLP approaches are shifting from pipeline architectures to an end-to-end ones. Some modern NLP solutions are implemented as end-to-end neuronal machine learning systems, such that the entire system is trained as a whole from the processing of input text to the output predication, and evaluation is to be done only on the final outputs.

Overall, system's architectures play a key role in deciding what to evaluate and should be taken into consideration when designing an evaluation methodology.

4.2 How to evaluate?

NLP systems are can be manually or automatically evaluated. Nowadays, manual evaluation in NLP are not uncommon. They usually require a well designed methodology, large investment in resources, and the results can be inconsistent and slow. Automatic evaluation is the most popular way of evaluating NLP systems, and they is usually designed to mimic human assessments. It require the creation of evaluation material (or gold-standard) and its cost depends on the complexity of the task to evaluate. It allows fast development and is cheap, in the sense that it is possible to re-use the evaluation material multiple times and sometimes it is possible to automate its creation process. As already mentioned, one of the main bottlenecks for advancing research in normative rules extraction is the lack of annotated gold-standard material for evaluation. Creating gold-standards can be an immense task, and its creation process needs to be scrutinised and evaluated to ensure certain quality. For example, it is expected that the same statement is to be annotated and agreed by at least two independent annotators, and inter-annotator agreement calculation, upper bounds discovery, etc., has to be done accordingly to measure the acceptability of the proposing standard. In the following, we provide some guidelines on creating a gold-standard for normative rules extraction.

4.2.1 How to Create a Gold-standard for Normative Rules Extraction

Having a gold-standard for normative rules extraction allows to carry-out intrinsic evaluations. Creating a gold-standard implies the compilation of a data set consisting of sentences written in natural language paired with their corresponding normative rules. With a gold-standard such as that, it is possible to perform evaluations in terms of *one input, one output*, and therefore, assess the quality of the normative rules directly, without having to execute the reasoning process.

Creating a gold-standard is can immense task. In what follows we discussed some important considerations to take into account when building a gold-standard data set for normative rules extraction.

- Agnostic to reasoning: there is no consensus about to what extent normative rules should be formalism independent, and whether normative rules should be agnostic to the reasoning machinery. In an ideal situation, the output of the normative mining algorithms is agnostic to the reasoning machinery that used them, which implies that the normative rules generated should be written in a format that can be transformed into a formalism that can be reasoned with at the later stage of the process.

- Annotator expertise level: the manual annotation of normative rules require humans with experience in the areas of logic, and/or linguistics, and/or law.

- Inter-annotator agreement and upper-bound calculation: Inter-annotator agreement is a measure of how well two (or more) annotators can make the same annotation decision for a certain instance. Calculating inter-annotator agreement gives an idea about the difficulty of the problem, thus how trustworthy is the annotation (the lower the agreement, the less trust-able the annotation) The agreement rate can be thought of as an upper bound (human ceiling), on accuracy of a system evaluated using the annotation.

- Gold-standard size and instance diversity: machine learning based approaches require about 1000 instances to learn a decent performing model. Meanwhile, other approaches might not require big gold-standard material for training, they should use gold-standards for development and evaluation with at least 100 instances. Making a gold-standard representative of a problem is hard, however, diversity in terms of rules complexity, regulation topics (rules from many different, unrelated regulations), and jurisdictions are likely to make a gold-standard more representative.

- Annotation of Deontic Modalities : it is necessary to characterise normative rules legally by attaching Deontic modalities to them.

- Annotation guidelines: annotation guidelines are useful to document conventions in annotation and should contemplate solutions for generic cases, for example, how to annotate negation, coordinate constructions, prepositional verbs (verbs coupled with specific prepositions e.g., *correlate with*), prepositional phrases as modifiers (prepositions are context dependent e.g., *corresponds with*, *corresponds to*) Annotation guidelines are important to help annotators to produce consistent annotations and help users to interpret the data correctly.

4.3 How to Evaluate Given the Gold-standard?

Having a gold standard means there is a reference for evaluation. As mentioned, a gold standard for normative rules consist of a set of sentences aligned with their normative rules. During evaluation, sentences from the gold standard are input to a computational model for rule extraction, which outputs normative rules.

Since normative rules are statements executable by a machine by a reasoning engine, they quality of the rules directly impact the reasoning process. Therefore, rules are either right or wrong, and there is no middle ground or partially correct rules. Hence, the most appropriate evaluation metric to assess the correctness of the rules is Accuracy.

Formally, accuracy is the fraction of predictions a model got right and it is defined as follows:

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{1}$$

The calculation of the predicted rules against the gold standard ones can be done by comparing the tokens of both rules sequentially from left to right. While the sequence of tokens is the same in both rules Accuracy is 1, otherwise is 0.

4.4 Road map for Evaluation and Benchmarking

Here we summarised key aspects towards the design of meaningful evaluations for normative rules extraction methods, and highlight some good practises to encourage methods comparison and facilitate replication.

Encourage *one input, one output intrinsic* evaluations: intrinsic evaluations focus on directly evaluating a desire functionality, for example, the correctness of the a normative rule given a sentence in natural language. This makes development more focus and faster than extrinsic evaluations, making easier to optimise the number of correct rules, thus minimising the risk of having multiple competing rules that can negatively affect the reasoning process.

Data sets creation: data sets creation is essential for evaluating methods for normative rules extraction. Using crowdsourcing platforms, which offers online workers with several levels of expertise, are probable the fasted way of annotating large amounts of data. For example, it is possible to crowdsource the construction of a data set of sentences paired with they corresponding normative rules. Performing a quality control it is important when eliciting annotations from online workers.

Data sets partitioning: data set partitioning is fundamental for reporting meaningful evaluations. Data sets should be partitioned into, at least, two disjoint subsets: test set and development set, so that the data in the test set a remain completely untouched and unseen until the system is frozen just prior to evaluation.

Data sets sharing: data sharing provides others with access to data. It avoids the generation of equivalent data sets, brings new perspectives from the re-analysis of the data set, and make possible method comparison and experiment replication.

Code sharing: code sharing provides other with access to implementation details and code of an existing method. Code sharing under open source licenses has become an standard good practise in computer science research. It avoids re-implementation of existing methods, improves their understanding, and facilitates method comparison.

5 Neuronal Semantic Parsing for Normative Rule Extraction

In this section, we assess the feasibility of using neuronal semantic parsing for the extraction of normative rules. To do so, we applied and evaluate an state-of-the-art neuronal semantic parsing approach using a small dataset of sentences from regulations and their corresponding representation in lambda calculus.

As discussed in Section 3, current NLP tools cannot be used to directly distil normative rules. Nevertheless, there are some promising avenues in that direction such as applying neuronal semantic parsing.

Semantic parsing is the task of mapping sentences in natural language to a meaning representation such as a logic formula (see Section 3.1). It can be seen as an intermediate step towards the extraction of normative rules due to its predicate-argument structure, which can be used as building blocks for normative rules extraction. Table 3 shows some example sentences written in natural language, their correspondent logic formula in lambda calculus (we follow the notation from [46]), and their corresponding normative rules represented using PCL [33].

We chose to evaluate neuronal semantic parsing since it does not require handcrafted features that usually require knowledge from domain experts in their design. In addition, it does not use syntactic structures to represent sentences, which can have poor quality when dealing with long sentences and domain-specific text, such as the legal one. Finally, neural network architectures are now the dominant machine

	Example
Sentence	*A large building is any building with a net lettable area greater than* $300\,\mathrm{m}^2$.
Logic formula	`lambda. $0 (if (A large building: $0) then (is any building with a lettable area greater than ($0 300m2)))`
Normative rule	$largeBuilding \rightarrow greaterThan(netLettableArea, 300)$
Sentence	*For the purposes of subclause 2.4, a person is responsible for an individual if the person is a parent of the individual.*
Logic formula	`lambda. $0 $1 (if (and (person:$0) (individual: $1) (parent of ($0 $1))) then (responsible for (for purpose of subclause 2.4 ($0 $1))))`
Normative rule	$subclause(2.4), parentOf(A, B) \Rightarrow_O responsible(A, B)$

Table 3: Examples of semantic parsing as an intermediate steps towards the generation of normative rules

learning approaches in NLP and produce the state-of-the-art results in semantic parsing.

In our experiments, we evaluate sequence-to-sequence model with LSTM [35] with Attention proposed by [21], as depicted in Figure 7. It consists of an encoder and decoder with two different L-layer recurrent neural networks with LSTM units, which recursively process tokens one by one. A sentence in natural language x is encoded into a vector representation, and decoded into a sequence $y_1, \ldots, y_{|y|}$ that is learned conditioned on the encoded vector $p(y|x)$. Additionally, to integrate encoder side information (also referred to as context vector), this approach incorporates an Attention Mechanism [5, 53].

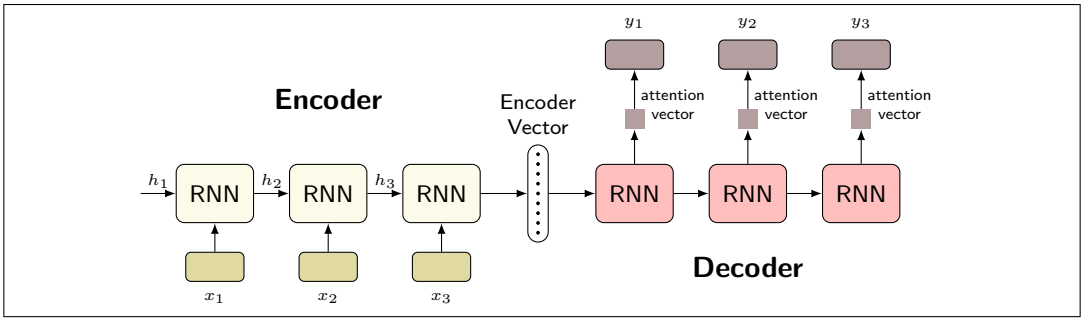


Figure 7: Sequence-to-Sequence Model with LSTM and Attention Architecture

	GEO	RegTech
Training set	600	144
Test set	280	-
Total	880	144
Avg. input length	7.3	26.75

Table 4: Data sets splits and average length of input and output sequences

Data set	Example
GEO	*what is the capital of the state with the largest population density?* `( capital:c ( argmax $1 ( state:t $1 ) ( density:i $1 ) ) )`
RegTech	*a large building is any building with a net lettable area greater than 300 m2.* `lambda. $0 (if (A large building:$0) then (is any building with a lettable area greater than ( $0 300m2)))`

Table 5: Examples of sentences written in natural language and their corresponding meaning representation from three data sets

5.1 Data Sets for Training Semantic Parsers

The neuronal semantic models are trained using the following data sets, which cover texts from different domains and their corresponding meaning representations is in lambda calculus.

GEO This is a standard semantic parsing benchmark. It consist of a set of queries to a database of U.S. geography. We used the splits provided by [21] (see Table 4). The meaning representation of this data set is lambda-calculus. Values for variables city, state, country, river and number are identified beforehand.

RegTech [27] This data set is developed for evaluating the performance of semantic parsing in the legal domain. At the moment, the data set consists of 140 sentences extracted from regulations from New Zealand and Australia. Following the annotation schema of [21], sentences in the data set are paired with logical expressions that are used to indicate the scope of the predicates and their arguments. The annotations were carried out by annotators with a background in logic. Each annotator annotated a set of 10 sentences (without overlap). Next, two annotators reviewed the logical expressions, agreed on the best practises and produce a consistent final version of the data set.

Information and examples of the data sets are shown in Tables 4 and 5, respectively.

Gold-standard formulas

1. `lambda $0 $1 $2 (and ((endorsing body or) (supplier of:$1))`
`then (must (be replaced by ($0 $1)))`
2. `lambda $0 (and ((claim:$0)) (must (not (refer to`
`($0 prevention of (or ((disease) (disorder) (condition))))))))`

Predicated formulas

1. `lambda $0 $1 $2 (and (( endorsing body:$0 )`
`(supplier of:$1 ( food:$2)) (must ( <U> ( $0 $1 )))))`
2. `lambda $0 (and((claim:$0)) (must (not (refer to`
`($0 diagnosis of (or (( disease) (disorder) (condition))))))))`

Table 6: Examples of gold-standard formulas from RegTech and their corresponding predicted formulas generated by the semantic parsing model

5.2 Experimental Settings

Semantic parsing is evaluated on accuracy, which is defined as the proportion of the input sentences that are correctly parsed to their gold standard logical form. All models are trained on GPU with their default hyper-parameters.

5.3 Experimental Results

Table 7 shows the results of the semantic parsing experiments. We first verify the ability of the method in analysing sentences of different complexity, assuming the sentence length is an indicator of sentence complexity (the longer the sentence, the more likely it is to contain complicated semantic structures). As already shown in Table 5, sentences in the legal domain (for example, in RegTech) have an average input length that is significantly higher than sentences in standard semantic parsing

	seq2seq+LSTM+Attention
GEO (all)	83.57
GEO (< 10 tokens)	92.09
GEO (> 10 tokens)	68.93
GEO (140)	29.28
RegTech (140)	18.28

Table 7: Semantic parsing evaluation on test sets (short sentences contained less than 0 tokens (< 10 tokens) and long sentences contained more than 10 tokens (> 10 tokens))

data sets such as GEO. Since the size RegTech is small and potentially not sufficient to properly train a semantic parser, we evaluate the parsers performance on long sentences by splitting the GEO test set in two subsets, respectively: a set containing sentences with less than 10 tokens; and a set containing sentences with more than 10 tokens. For comparison, we also report the evaluation results with the full test sets: GEO (all). As expected, results shows considerable drops when parsing long sentences, The model performance dropped about 14 points when parsing long sentences with GEO (from 83.57 to 68.99).

As mentioned, the size of RegTech is potentially too small to train a semantic parser. Results shows an accuracy of 18.28 for sequence-to-sequence. For comparison, we report results on GEO trained only with 140 sentences that were randomly chosen. Results indicate that models trained with limited amount of data are not able to generalise well. A qualitative error analysis performed on the output of the semantic parsers trained with RegTech indicates that the models are able to correctly output the structure of the logic formulas, but failed to instantiate the appropriate vocabulary. Cf. Table 6. We attribute this limitation to the vocabulary mismatch between the training and testing sets. The main take away from these experiments is that current technologies for semantic parsing are data hungry, and creating data sets for semantic parsing is not a trivial task. As mentioned before, in the GEO data set values for in-domain variables e.g., city, airport, etc. are anonymised before training, thus the vocabulary size is reduced, making encoding and decoding easier. Note that variables in RegTech are not anonymised, consequently, the vocabulary size is bigger, which impacts the generalisation power of the model. Nevertheless, we argue that is less costly to increase the size of RegTech and train a semantic parser, than investing in a syntax-based approach, which require to manually annotate in-domain syntactic structures to re-train a syntax analyser and to write grammars to distil the rules from the trees.

6 Conclusion

In this chapter, we have identified some of the problems of extracting normative rules from legal texts from an NLP perspective. One of the main challenges is the semantic analysis of sentences such as the identification of predicates and their arguments in complex sentences. Other important aspects to take into account in the design and development of methods for normative rule extraction are: cross-referencing between document sections and document collections; the identification of normative effects or normative modalities; and the intrinsic ambiguities that might arise as natural language is use to write legislation and regulations.

Despite the difficulties of extracting normative rules from text, several approaches have been developed with certain success. While syntax driven approaches are the most popular strategy for normative rule extraction, the current limitations of syntax parsing in the legal domain make these approaches insufficient. The most obvious limitation of syntactic analysis is related to structural ambiguity, which occurs when a parser produces a competitive analysis, hence producing more than one possible parsed outcome to a sentence. Another limitation of the current state-of-the-art approaches is the lack of an standard evaluation methodology. The lack of publicly available data sets for evaluation makes model comparison and replication not possible. We argue that to overcome the evaluation issues just mentioned and to advance research in this area it is necessary to encourage researchers to share their data and models.

This chapter also includes experiments on neuronal semantic parsing as an intermediate step towards the extraction of normative rules. In our experiments, we have used a publicly available data set of sentences from legal documents aligned with lambda calculus formulas, which is the only resource available for semantic parsing in the legal domain. We believed this is an promising research avenue as neuronal models for semantic parsing are receiving a lot of attention from the NLP and machine learning community.

The field of NLP is moving fast, and it is a challenge for the legal informatics community to keep up with the advances. Therefore, encouraging multi-disciplinary teams with NLP researchers and legal informatics experts is the key to advance research in this area.

References

[1] Gianmaria Ajani, Guido Boella, L. Lesmo, Macro Martin, Alessandro Mazzei, and Piercarlo Rossi. A Development Tool For Multilingual Ontology-based Conceptual Dictionaries. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, LREC06, pages 479–484. Genoa, Italy, May 2006.

[2] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. Leveraging Linguistic Structure For Open Domain Information Extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354. Association for Computational Linguistics, Beijing, China, July 2015.

[3] Alain Auger and Caroline Barrière. Pattern-based approaches to semantic relation extraction: A state-of-the-art. *Terminology*, 14(1):1–19, January 2008. ISSN 0929-9971. URL `https://www.jbe-platform.com/content/journals/10.1075/term.14.1.02aug`.

[4] C. J. W. Baaij. *Legal Integration and Language Diversity: Rethinking Translation in EU Lawmaking*. Oxford University Press, 2018.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[6] Martina Bajčić. The Role of EU Legal English in Shaping EU Legal Culture. *International Journal of Language & Law*, 7:8–24, 2018.

[7] Imran Sarwar Bajwa, Mark G. Lee, and Behzad Bordbar. SBVR Business Rules Generation from Natural Language Specification. In Knut Hinkelmann and Barbara Thönssen, editors, *2011 AAAI Spring Symposium: AI for Business Agility*, pages 2–8, 2011.

[8] V. Richard Benjamins, Pompeu Casanovas, Joost Breuker, and Aldo Gangemi, editors. *Law and the Semantic Web: Legal Ontologies, Methodologies, Legal Information Retrieval, and Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. 1–17 pp.

[9] C. Biagioli, E. Francesconi, A. Passerini, S. Montemagni, and C. Soria. Automatic Semantics Extraction in Law Documents. In *Proceedings of the 10th International Conference on Artificial Intelligence and Law*, ICAIL '05, pages 133–140. ACM, Bologna, Italy, June 2005.

[10] Guido Boella, Luigi Di Caro, and Livio Robaldo. Semantic Relation Extraction from Legislative Text Using Generalized Syntactic Dependencies and Support Vector Machines. In Leora Morgenstern, Petros Stefaneas, François Lévy, Adam Wyner, and Adrian Paschke, editors, *Proceedings of the 7th International Web Rule Symposium: Theory, Practice, and Applications of Rules on the Web*, RuleML 2013, pages 218–225. Springer Berlin Heidelberg, Seattle, WA, USA, 2013.

[11] Alexander Boer, Rinke Hoekstra, and Radboud Winkels. METALex: Legislation in XML. In T. Bench-Capon, A. Daskalopulu, and R. Winkels, editors, *Proceedings of the 15th Annual Conference on Legal Knowledge and Information Sytems*, JURIX 2002, pages 1–10. IOS Press, 2002.

[12] T.D. Breaux and A.I. Anton. Analyzing Regulatory Rules for Privacy and Security Requirements. *IEEE Transactions on Software Engineering*, 34(1):5–20, January 2008. ISSN 0098-5589.

[13] Travis D. Breaux and Annie I. Antón. A Systematic Method for Acquiring Regulatory Requirements: A Frame-Based Approach. In *Proceedings of the 6th International Workshop on Requirements for High Assurance Systems*, RHAS-6. Software Engineering Institute (SEI), Pittsburg, PA, USA, September 2007.

[14] Travis Durand Breaux. *Legal Requirements Acquisition for the Specification of Legally Compliant Information Systems*. Ph.D. Thesis, North Carolina State University, Raleigh, NC, USA, 2009.

[15] Alicia Burga, Joan Codina, Gabriela Ferraro, Horacio Saggion, and Leo Wanner. The Challenge of Syntactic Dependency Parsing Adaptation for the Patent Domain. In *Proceedings of ESSLLI-13 Workshop on Extrinsic Parse Improvement*, EPI. Düsseldorf,

Germany, August 2013.

[16] Elena Cabrio and Serena Villata. Natural Language Arguments: A Combined Approach. In *Proceedings of the 20th European Conference on Artificial Intelligence*, ECAI'12, pages 205–210. IOS Press, Montpellier, France, August 2012.

[17] Elena Cabrio and Serena Villata. Five Years of Argument Mining: a Data-driven Analysis. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI 2018, pages 5427–5433. International Joint Conferences on Artificial Intelligence Organization, July 2018.

[18] Hamish Cunningham, Valentin Tablan, Angus Roberts, and Kalina Bontcheva. Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. *PLOS Computational Biology*, 9(2):1–16, 02 2013.

[19] Denis A. de Araujo, Sandro J. Rigo, Carolina Muller, and Rove Chishman. Automatic Information Extraction from Texts with Inference and Linguistic Knowledge Acquisition Rules. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 3, pages 151–154. IEEE, Atlanta, GA, USA, November 2013.

[20] Emile de Maat, Radboud Winkels, and Tom van Engers. Automated Detection of Reference Structures in Law. In Tom M. van Engers, editor, *Proceedings of the 19th International Conference on Legal Knowledge and Information Systems*, JURIX 2006, pages 41–50. IOS Press, Amsterdam, The Netherlands, December 2006.

[21] Li Dong and Mirella Lapata. Language to Logical Form with Neural Attention. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2016, pages 33–43. Association for Computational Linguistics, Berlin, Germany, August 2016.

[22] 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 International Workshops*, AICOL 2015, pages 287–300. Springer International Publishing, 2015.

[23] Mauro Dragoni, Serena Villata, Williams Rizzi, and Guido Governatori. Combining NLP Approaches for Rule Extraction from Legal Documents. In Ugo Pagallo, Monica Palmirani, Pompeu Casanovas, Giovanni Sartor, and Serena Villata, editors, *1st Workshop on MIning and REasoning with Legal texts*, MIREL 2016. Sophia Antipolis, France, December 2016.

[24] Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. Multilingual Semantic Parsing And Code-Switching. In Roger Levy and Lucia Specia, editors, *Proceedings of the 21st Conference on Computational Natural Language Learning*, CoNLL 2017, pages 379–389. Association for Computational Linguistics, Vancouver, Canada, August 2017.

[25] Tatjana Evas. EU law for an open independent and efficient European administration: Summary report of the public consultation. Study PE 621.830, European Added Value Unit, European Parliament Research Service, July 2018. URL `http://www.europarl.`

europa.eu/thinktank/en/document.html?reference=EPRS_STU(2018)621830.

[26] Davide Falessi, Giovanni Cantone, and Gerardo Canfora. A Comprehensive Characterization of NLP Techniques for Identifying Equivalent Requirements. In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '10, pages 18:1–18:10. ACM, Bolzano-Bozen, Italy, September 2010.

[27] Gabriela Ferraro, Ho-Pun Lam, Silvano Colombo Tosatto, Francesco Oliveri, Nick van Beest, and Guido Governatori. Automatic Extraction of Legal Norms: Evaluation of Natural Language Processing Tools. In *Proceedings of the 13th International Workshop on Juris-Informatics*, JURISIN 2019. Springer, Kanagawa, Japan, November 2019.

[28] E. Francesconi and A. Passerini. Automatic Classification of Provisions in Legislative Texts. *Artificial Intelligence and Law*, 15(1):1–17, March 01, 2007. ISSN 1572-8382.

[29] Shruti Gaur, Nguyen H. Vo, Kazuaki Kashihara, and Chitta Baral. Translating Simple Legal Text to Formal Representations. In Tsuyoshi Murata, Koji Mineshima, and Daisuke Bekki, editors, *8th International Workshop on Juris-Informatics*, JURISIN 2014, pages 259–273. Springer Berlin Heidelberg, Kanagawa, Japan, October 2014.

[30] Ruifang Ge and Raymond Mooney. A Statistical Semantic Parser that Integrates Syntax and Semantics. In Ido Dagan and Daniel Gildea, editors, *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CoNLL-2005, pages 9–16. Association for Computational Linguistics, Ann Arbor, MI, USA, June 2005.

[31] Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho. *Ontological Engineering: with Examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. Springer, London, UK, 2004. 1–45 pp.

[32] David G. Gordon and Travis D. Breaux. A cross-domain empirical study and legal evaluation of the requirements water marking method. *Requirements Engineering*, 18 (2):147–173, June 2013.

[33] Guido Governatori and Antonino Rotolo. A Conceptually Rich Model of Business Process Compliance. In *Proceedings of the 7th Asia-Pacific Conference on Conceptual Modelling*, APCCM 2010, pages 3–12. ACS, Brisbane, QLD, Australia, January 2010.

[34] Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. Syntax for Semantic Role Labeling, To Be, Or Not To Be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2061–2071. Association for Computational Linguistics, Melbourne, Australia, July 2018.

[35] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computing*, 9(8):1735–1780, November 1997.

[36] Rinke Hoekstra, Joost Breuker, Marcello Di Bello, and Alexander Boer. The LKIF Core Ontology of Basic Legal Concepts. In Pompeu Casanovas, Maria Angela Biasiotti, Enrico Francesconi, and Maria Teresa Sagri, editors, *Proceedings of the Workshop on Legal Ontologies and Artificial Intelligence Techniques*, LOAIT 2007. Stanford, CA, USA, June 2007.

[37] Theo M. V. Janssen. Compositionality – with an appendix by b. partee, 1997.

[38] Robin Jia and Percy Liang. Data Recombination for Neural Semantic Parsing. In Katrin Erka and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2016, pages 12–22. Association for Computational Linguistics, Berlin, Germany, August 2016.

[39] Xing Jiang and Ah-Hwee Tan. Mining ontological knowledge from domain-specific text documents. In *Proceedings of the 5th IEEE International Conference on Data Mining*, ICDM'05. IEEE, Houston, TX, USA, November 2005.

[40] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice-Hall, Upper Saddle River, NJ, USA, 2nd edition, 2008.

[41] Nal Kalchbrenner and Phil Blunsom. Recurrent Continuous Translation Models. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2013, pages 1700–1709. Association for Computational Linguistics, Seattle, Washington, USA, October 2013.

[42] Rohit J. Kate and Raymond J. Mooney. Using String-Kernels for Learning Semantic Parsers. In Rohit J. Kate and Raymond J. Mooney, editors, *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 913–920. Association for Computational Linguistics, Sydney, NSW, Australia, July 2006.

[43] Yusuke Kimura, Makoto Nakamura, and Akira Shimazu. Treatment of Legal Sentences Including Itemized and Referential Expressions – Towards Translation into Logical Forms. In Hiromitsu Hattori, Takahiro Kawamura, Tsuyoshi Idé, Makoto Yokoo, and Yohei Murakami, editors, *Proceedings of the 22nd Annual Conference of the JSAI: New Frontiers in Artificial Intelligence*, JSAI 2008, pages 242–253. Springer Berlin Heidelberg, Asahikawa, Japan, June 2009.

[44] Nadzeya Kiyavitskaya, Nicola Zeni, Travis D. Breaux, Annie I. Antón, James R. Cordy, Luisa Mich, and John Mylopoulos. Automating the Extraction of Rights and Obligations for Regulatory Compliance. In Qing Li, Stefano Spaccapietra, Eric Yu, and Antoni Olivé, editors, *Proceedings of the 27th International Conference on Conceptual Modeling*, ER 2008, pages 154–168. Springer Berlin Heidelberg, Barcelona, Spain, October 2008.

[45] Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. Scaling Semantic Parsers with On-the-Fly Ontology Matching. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2013, pages 1545–1556. Association for Computational Linguistics, Seattle, WA, USA, October 2013.

[46] Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Lexical Generalization in CCG Grammar Induction for Semantic Parsing. In Regina Barzilay and Mark Johnsn, editors, *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2011, pages 1512–1523. Association for Computational Linguistics, Edinburgh, Scotland, UK, July 2011.

[47] Mirella Lapata and Li Dong. Coarse-to-Fine Decoding for Neural Semantic Parsing. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2018, pages 731–742. Association for Computational Linguistics, Melbourne, Australia, July 2018.

[48] Seok-Won Lee, Robin Gandhi, Divya Muthurajan, Deepak Yavagal, and Gail-Joon Ahn. Building Problem Domain Ontology from Security Requirements in Regulatory Documents. In *Proceedings of the 2006 International Workshop on Software Engineering for Secure Systems*, SESS '06, pages 43–50. ACM, Shanghai, China, May 2006.

[49] Percy Liang, Michael I. Jordan, and Dan Klein. Learning Dependency-based Compositional Semantics. *Comput. Linguist.*, 39(2):389–446, June 2013. ISSN 0891-2017.

[50] Wang Ling, Phil Blunsom, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, Fumin Wang, and Andrew Senior. Latent Predictor Networks for Code Generation. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2016, pages 599–609. Association for Computational Linguistics, Berlin, Germany, August 2016.

[51] Marco Lippi and Paolo Torroni. Argumentation Mining: State of the Art and Emerging Trends. *ACM Transactions on Internet Technology*, 16(2):10:1–10:25, March 2016.

[52] Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. A Generative Model for Parsing Natural Language to Meaning Representations. In Mirella Lapata and Hwee Tou Ng, editors, *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2008, pages 783–792. Association for Computational Linguistics, Honolulu, HI, USA, October 2008.

[53] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421. Association for Computational Linguistics, Lisbon, Portugal, September 2015.

[54] Jeremy C. Maxwell, Annie I. Antón, and Peter Swire. A Legal Cross-References Taxonomy for Identifying Conflicting Software Requirements. In *2011 IEEE 19th International Requirements Engineering Conference*, RE 2011, pages 197–206. IEEE, Trento, Italy, August 2011.

[55] Hugh Miall. *Emergent Conflict and Peaceful Change*. Palgrave MacMillan, 2007.

[56] Ministry of Business, Innovation and Employment. *New Zealand Building Code Clause G4 Ventilation: Acceptable Solutions and Verification Methods*. New Zealand Government, New Zealand, 4th edition, 2017.

[57] Ministry of Business, Innovation and Employment. *New Zealand Building Code Clause H1 Energy Efficiency: Acceptable Solutions and Verification Methods*. New Zealand Government, New Zealand, 4th edition, 2017.

[58] Janardan Misra. Terminological inconsistency analysis of natural language requirements. *Information and Software Technology*, 74:183 – 193, 2016.

[59] Raquel Mochales-Palau and Marie-Francine Moens. Argumentation Mining: The Detection, Classification and Structure of Arguments in Text. In *Proceedings of the 12th*

International Conference on Artificial Intelligence and Law, ICAIL '09, pages 98–107. ACM, Barcelona, Spain, June 2009.

[60] Raquel Mochales-Palau and Marie-Francine Moens. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22, March 01, 2011. ISSN 1572-8382.

[61] Richard Montague. The proper treatment of quantification in ordinary English. In K. J. J. Hintikka, J. Moravcsic, and P. Suppes, editors, *Approaches to Natural Language*, pages 221–242. Reidel, Dordrecht, 1973.

[62] Makoto Nakamura, Shunsuke Nobuoka, and Akira Shimazu. Towards Translation of Legal Sentences into Logical Forms. In Ken Satoh, Akihiro Inokuchi, Katashi Nagao, and Takahiro Kawamura, editors, *Proceedings of the 21st Annual Conference (and Workshops) of the Japanese Society for Artificial Intelligence*, JSAI 2007, pages 349–362. Springer Berlin Heidelberg, Miyazaki, Japan, June 2008.

[63] Object Management Group. *Semantics of Business Vocabulary and Rules (SBVR)*, 1.4 edition, May 2017. URL `https://www.omg.org/spec/SBVR/`. [accessed: 1 August 2019].

[64] Monica Palmirani, Raffaella Brighi, and Matteo Massini. Automated Extraction of Normative References in Legal Texts. In *Proceedings of the 9th International Conference on Artificial Intelligence and Law*, ICAIL '03, pages 105–106. ACM, Scotland, UK, June 2003. ISBN 1-58113-747-8.

[65] Monica Palmirani and Fabio Vitali. *OASIS LegalDocumentLegal (LegalDocML)*, 2018. 3–12 pp. URL `http://docs.oasis-open.org/legaldocml/akn-core/v1.0/akn-core-v1.0-part1-vocabulary.html`.

[66] Sachin Pawar, Girish K. Palshikar, and Pushpak Bhattacharyya. Relation Extraction : A Survey. *CoRR*, abs/1712.05191, 2017. URL `http://arxiv.org/abs/1712.05191`.

[67] Roger D. Peng. Reproducible research in computational science. *Science*, 334, 12 2011.

[68] Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. Transforming Dependency Structures to Logical Forms for Semantic Parsing. *Transactions of the Association for Computational Linguistics*, 4:127–141, 2016.

[69] Rossella Rubino, Antonino Rotolo, and Giovanni Sartor. An OWL Ontology of Fundamental Legal Concepts. In *Proceedings of the 9th Annual Conference on Legal Knowledge and Information Systems*, JURIX 2006, pages 101–110. IOS Press, Amsterdam, The Netherlands, 2006.

[70] Matt Selway, Georg Grossmann, Wolfgang Mayer, and Markus Stumptner. Formalising natural language specifications using a cognitive linguistic/configuration based approach. *Information Systems*, 54:191 – 208, 2015.

[71] Amin Sleimi, Nicolas Sannier, Mehrdad Sabetzadeh, Lionel Briand, and John Dann. Automated Extraction of Semantic Legal Metadata Using Natural Language Processing. In *The 26th IEEE International Requirements Engineering Conference*, pages 124–135. IEEE, Banff, AB, Canada, August 2018.

[72] Christian Stab and Iryna Gurevych. Parsing Argumentation Structures in Persuasive

Essays. *Computational Linguistics*, 43(3):619–659, 2017.

[73] Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. Improving Machine Reading Comprehension with General Reading Strategies. In Jull Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL 2019, pages 2633–2643. Association for Computational Linguistics, Minneapolis, MN, USA, June 2018.

[74] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. *CoRR*, abs/1409.3215, 2014.

[75] Oanh Thi Tran, Minh Le Nguyen, and Akira Shimazu. Reference Resolution in Legal Texts. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law*, ICAIL '13, pages 101–110. ACM, Rome, Italy, June 2013.

[76] Tom M. van Engers, Ron van Gog, and Kamal Sayah. A Case Study on Automated Norm Extraction. In Thomas Gordon, editor, *The 17th International Conference on Legal Knowledge and Information Systems*, JURIX 2004, pages 49–58. IOS Press, Amsterdam, The Netherlands, 2004.

[77] Ron van Gog and Tom M. van Engers. Modeling legislation using natural language processing. In *2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace (Cat.No.01CH37236)*, volume 1, pages 561–566. IEEE, Tucson, AZ, USA, October 2001.

[78] Pepijn R. S. Visser and Trevor J. M. Bench-Capon. A Comparison of Four Ontologies for the Design of Legal Knowledge Systems. *Artificial Intelligence and Law*, 6(1):27–57, March 1998. ISSN 1572-8382.

[79] David H. D. Warren and Fernando C. N. Pereira. An Efficient Easily Adaptable System for Interpreting Natural Language Queries. *Computational Linguistics*, 8(3-4):110–122, July 1982. ISSN 0891-2017.

[80] Radboud Winkels and Rinke Hoekstra. Automatic Extraction of Legal Concepts and Definitions. In Burkhard Schäfer, editor, *The 25th International Conference on Legal Knowledge and Information Systems*, JURIX 2012, pages 157–166. IOS Press, Amsterdam, The Netherlands, December 2012.

[81] Kristian Woodsend and Mirella Lapata. WikiSimple: Automatic Simplification of Wikipedia Articles. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, AAAI 2011, pages 927–932. AAAI Press, San Francisco, CA, USA, August 2011.

[82] Adam Z. Wyner and Wim Peters. On rule extraction from regulations. In *International Conference on Legal Knowledge and Information Systems, JURIX*, 2011.

[83] John M. Zelle and Raymond J. Mooney. Learning to Parse Database Queries Using Inductive Logic Programming. In William J. Clancey and Daniel S. Weld, editors, *Proceedings of the 13th National Conference on Artificial Intelligence*, volume 2, pages 1050–1055. AAAI Press / The MIT Press, Portland, Oregon, USA, August 1996.

[84] Luke S. Zettlemoyer and Michael Collins. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *Proceedings of*

the Twenty-First Conference on Uncertainty in Artificial Intelligence, UAI'05, pages 658–666. AUAI Press, Arlington, VA, USA, 2005.

[85] George K. Zipf. *Human Behavior and the Principle of Least Effort.* Addison-Wesley, 1949.

[86] Alessandro Zucchi. *The Language of Propositions and Events: Issues in the Syntax and the Semantics of Nominalization.* Springer Netherlands, Dordrecht, Germany, 1993. 1–32 pp.

Received 1 February 2021

Textual Entailment for Cybersecurity: An Applicative Case

Giovanni Siragusa
University of Turin, Turin, Italy
siragusa@di.unito.it

Livio Robaldo
Legal Innovation Lab Wales, Swansea University, United Kingdom
Nomotika SRL, Turin, Italy
livio.robaldo@gmail.com

Luigi Di Caro
University of Turin, Turin, Italy
Nomotika SRL, Turin, Italy
dicaro@di.unito.it

Andrea Violato
Nomotika SRL, Turin, Italy
andrea.violato@nomotika.it

Abstract

Recognizing Textual Entailment (RTE) is the task of recognizing the relation between two sentences, in order to measure whether and to what extent one of the two is inferred from the other. It is used in many Natural Language Processing (NLP) tasks. In the last decades, with the digitization of many legal documents, NLP applied to the legal domain has became prominent, due to the need of knowing which norms are complied with in case other norms are. In this context, from a set of obligations that are known to be complied with, RTE may be used to infer which other norms are complied with as well. We propose a dataset, regarding cybersecurity controls, for RTE on the legal domain. The dataset has been constructed using information available online, provided by domain experts from NIST (https://www.nist.gov).

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

1 Introduction

It is well-known that laws can be pragmatically interpreted in multiple, and often incompatible, ways, even in the same context. Handling multiple interpretations of legal norms is perhaps the best known problem in Legal Informatics.

On the one hand, since it is impossible to predict a priori every possible context where the norms will be deployed, legislators tend to use vague terms that are flexible enough to be adapted to a multitude of contexts and, within certain limits, to the technological advancements of the society.

On the other hand, what makes legal texts so much dependent on subjective human interpretation is that they are used in disputes that represent different interests, so that the interpretation of the norms tends to be stretched depending on the interest involved.

It is eventually up to judges and other appointed authorities to decide the interpretation of norms in context. According to the seminal work in [13], legal authorities expand or restrict the core of determinate meaning of norms by filling legal gaps to connect *legal requirements* (formal compliance) and *operational requirements* (substantive compliance), i.e., how and to what extent the legal requirements from legislation are met in real-world scenarios.

More generally, the connection from legal to operational requirements recalls the notion of "concept holism" (see [7, 11, 27, 3], among others): one cannot say to have the complete meaning of a legal requirement without knowing the whole system of constitutive rules and the web of concepts with which the meaning of that requirement is intertwined.

In order to take decisions about the interpretation of norms, judges often consult the relevant literature in the area. For this reason, other legal authorities, standardization bodies, or associations representing categories of involved entities produce additional documents that contain recommendations, guidelines, standards, etc. specifying how to be compliant with the legislation in specific situations. In many cases, this is even explicitly required by the legislation itself, as in the case of the General Data Protection Regulation (GDPR), which requires controllers to define their own data protection policies (cf. GDPR, Artt. 13, 14, and 24(2)), invites associations and other bodies representing categories of controllers or processors to prepare codes of conducts (see, e.g., GDPR, Art. 40), the European Data Protection Board has the duty to release guidelines and recommendations (see, e.g., GDPR, Art. 70(1)(d)), etc. See discussion in [22] and [25].

Recommendations, guidelines, standards, etc. are not typically part of legislation; therefore, their adoption do not automatically provide compliance with the regulations. However, by certifying the adoption of a standard, an organisation can

argue in favour of its proactive attitude and best efforts to be compliant with the regulations. In other words, such certifications provide strong arguments of compliance to be possibly used in auditing procedures or even in court.

On the other hand, since operational requirements are usually scattered around several documents in different format released, at different times, by different associations and other bodies, with different authoritative power and reputation in a certain domain, finding correlations between legal and operational requirements requires to build, maintain, and analyze an up-to-date archive of all these documents, which may be rather time-consuming, burdensome, and, therefore, unmanageable.

In light of this, Natural Language Processing (NLP) applications, in particular Textual Entailment (TE) applications [16], can provide valid help in creating and maintaining such an holistic network of legal/operational requirements, specifically in identifying when a requirement semantically entails another one.

The main problem of TE regarding legal documents is the availability of dataset used to train machine learning algorithms to recognize the relation expressed. The few existing ones are generally based on case laws, as the one proposed in Competition on Legal Information Extraction/Entailment (COLIEE) Workshop[1]. There is no dataset regarding standard procedures that a company has to implement to protect their data, where the adoption of TE techniques are crucial to verify if they have been defined.

Such procedures are generally defined by ISO[2] (the International Organization for Standardization). The standard includes 114 *controls*, that a company needs to check in order to consider itself as "secure" enough from cybersecurity attacks. In order to assess compliance with the standard, a company hires specialized auditors, who, after an inspection, decides whether the company is compliant with the standard or has to revise some of its internal business processes.

However, the ISO/IEC 27001:2013 controls, expressed in Natural Language, are quite vague and leave plenty of room for subjective interpretations.

For this reason, several public institutions, e.g., NIST[3] (National Institute of Standards and Technology), release more context-specific standards that refine the ISO/IEC 27001:2013. In this paper, we focus in particular on the NIST 800-53 rev.4[4], which implements 256 controls while specifying how they relate to the 114 controls of ISO/IEC 27001:2013 and viceversa. Specifically, it contains annexes that explain which controls of one of the two standards are satisfied by the controls of the other (and viceversa), in the sense that if a company implements one of the two,

[1] https://sites.ualberta.ca/~rabelo/COLIEE2019/

[2] https://www.iso.org/home.html

[3] https://www.nist.gov

[4] https://nvd.nist.gov/800-53

then it is assumed that the company also implements the associated controls in the other standard.

The present paper starts from the assumption that the ISO/IEC 27001:2013 and NIST 800-53 rev.4, and, in particular, the annexes included in the latter, which specify correspondences between the two standards, are precious raw sources for building a dataset for RTE. The latter has been recently identified in [4] as a challenging research topic.

Note that ISO/IEC 27001:2013 and NIST 800-53 rev.4 are just the two running examples that we will use in this paper. Many other cybersecurity standards are available on the Web, as well as corresponding tables inter-linking their controls. In other words, this paper has to be considered as the first step of a bigger research project to create a dataset made of a *network* of inter-connected technical documents in the cybersecurity domain. The advocated dataset, and the RTE classifiers trained, tuned, and evaluated on it, would be a precious resource for cybersecurity auditors and companies collaborating with them, e.g., Nomotika SRL.

In this paper, we propose:

- A dataset for RTE regarding cybersecurity. We constructed the dataset using the correspondences between controls that we found in the ISO/IEC 27001:2013 and NIST 800-53 rev.4.
- An evaluation of several RTE classifiers on the dataset, where we conducted a three-step evaluation. In the first step, we evaluated the dataset using cross-fold validation to see if it could be used to train the RTE classifiers. In the second step, we trained the classifiers using the COLIEE dataset[5] for Legal Textual Entailment; the idea is to check whether it is possible to transfer the knowledge acquired from a domain-oriented dataset to our one. Finally, in the third step, we checked if it is possible to transfer the knowledge acquired on our dataset to other legal ones.

The remainder of the paper is structured as follows: Section 2 describes the construction of the dataset, reporting the number of pairs it contains, the average length of sentences and the vocabulary size; Section 3 describes the used models and their performance on the two datasets. Section 4 describes some related works on legal domain and RTE. Section 5 concludes the paper.

[5] `https://sites.ualberta.ca/~rabelo/COLIEE2019/`

2 A dataset for Recognizing Textual Entailment (RTE)

We defined a dataset for RTE in the cybersecurity domain, called *cybersecurity entailment*. We constructed the dataset using ISO controls covered by the NIST ones[6], and the controls of the same NIST document[7] covered by the ISO/IEC 27001. For NIST and ISO documents, each <NIST control, ISO control> pair is constructed using the table[8] reported in the NIST document, and it could be seen as an entailment pair. We then extended the entailment pairs with neutral ones, applying a cartesian product between NIST and ISO controls and removing duplicated ones. An example of positive <NIST control, ISO control> pair follows:

NIST: "*The organization employs the principle of least privilege, allowing only authorized accesses for users (or processes acting on behalf of users) which are necessary to accomplish assigned tasks in accordance with organizational missions and business functions.*"

ISO: "*The allocation and use of privileged access rights shall be restricted and controlled.*"

The following one is an example of neutral <NIST control, ISO control> pair:

NIST: "*The information system enforces approved authorizations for logical access to information and system resources in accordance with applicable access control policies.*"

ISO: "*Users shall ensure that unattended equipment has appropriate protection.*"

We repeated the process of constructing the pairs for the controls between the NIST document and the ISO/IEC 27001. In this case, each ISO/IEC control has a *related_to* tag that expresses connections with other NIST ones, reporting their IDs. The IDs in each tag are assigned by a domain expert. We used the tag to construct the entailment pairs. We then extended the set with neutral pairs as for the previous set. A <NIST control, ISO/IEC control> pair that expresses an entailment relation follows:

sent.: *The organization implements a tamper protection program for the information system, system component, or information system service.*

[6] `https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf`, appendix H

[7] `https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf`, appendix F

[8] The table is created by a domain expert when the document is redacted.

related_to: *The organization protects against supply chain threats to the information system, system component, or information system service by employing as part of a comprehensive, defense-in-breadth information security strategy.*

The following pair is an example of neutral <NIST control, ISO/IEC control> pair:

sent.: *The information system maintains a separate execution domain for each executing process.*

related_to: *The information system separates user functionality (including user interface services) from information system management functionality.*

Finally, we merged the two sets of pairs to create the *cybersecurity entailment* dataset. We balanced the resulting dataset in order to have the same number of entailment and neutral pairs. An interesting fact is that the constructed dataset does not contain contradiction pairs because a control cannot be in contraposition with an another one.

Table 1 reports the number of pairs, the average sentence length, and the size of unique terms. The table highlights that the vocabulary of neutral pairs is contained in the entailment one. We also report the frequency of the Part-Of-Speech (POS) tags in Figure 1. We used OpenNLP[9] to assign the POS tags. From the image, it is possible to see that the majority of words are nouns, followed by adjectives.

The proposed dataset is availabled at `https://drive.google.com/drive/folders/1swYciO8yOtaM1pCTS9ySEpNZ-Ac8A569?usp=sharing`

	# pairs	avg. sentence length	vocabulary size
all dataset	2898	110.0	1912
entailment	1449	115.37	1912
neutral	1449	104.59	1905

Table 1: The table reports the number of pairs, the average sentence length, and the size of unique terms of the *cybersecurity entailment* dataset.

2.1 XML representation of the dataset

We stored the dataset into an XML file to simplify sharing and interoperability. We encapsulated each <premise, hypothesis> pair inside the *pair* tag. In each tag, the first element of the pair is contained in the *t* tag (the premise) and the second element in the *h* tag (the hypothesis). Furthermore, each *pair* tag has the attribute *entailment* which expresses the

[9] `https://opennlp.apache.org`

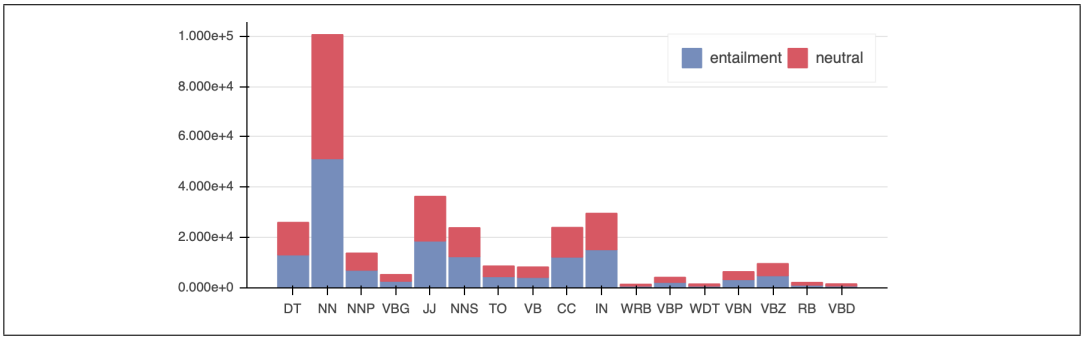


Figure 1: The POS frequency of the *cybersecurity entailment* dataset.

relation: entailment or neutral. It also has two other attributes: *id* which is an identifier of the pair, and *task* which is required by Excitement Open Platform (EOP) framework [17, 21]. All those entries are contained under the tag *entailment-corpus*. Figure 2 depicts an excerpt of the xml.

The main advantage of such XML structure is that it can be passed as an input to the EOP framework (or another one for TE) in order to train a classifier.

3 Evaluation Study of the Cybersecurity Dataset

In this section, we will perform three different analysis on our dataset:

- we will evaluate whether it is possible to train a classifier for RTE in order to recognize the entailment relation expressed in the <NIST, ISO> pair. For this evaluation, we will perform a cross-fold validation on our dataset;
- we will evaluate whether it is possible to transfer the knowledge acquired from another dataset to our own. In this evaluation, we would like to check the complexity of our dataset, i.e. if the relation expressed in the pairs can be easily recognized;
- we will evaluate whether it is possible to use our dataset to recognize the TE relations present in another dataset for legal domain, i.e. if a classifier trained on our dataset can generalize on unseen data. For this evaluation, we will train the classifiers on the *cybersecurity entailment* dataset and we will test them on the COLIEE testset.

.

We will follow RTE evaluation using *accuracy* measure: the ratio between the number of test instances correctly predicted and the number of test instances.

3.1 Preprocessing of the Sentences

In order to train the classifiers, we have to process the dataset sentences to extract the relevant features. The preprocessing consists of the following steps: tokenization, stopwords removal, stemming and feature extraction.

```
<?xml version="1.0" ?>
<entailment-corpus lang="EN">
  <pair entailment="ENTAILMENT" id="0" task="IR">
    <h>The organization retains audit records for [Assignment: organization-defined time period consistent with records retention policy] to provide support for after-the-fact investigations of
      security incidents and to meet regulatory and organizational information retention requirements.</h>
    <t>The information system provides the capability for authorized users to select a user session to capture/record or view/hear.</t>
  </pair>
  <pair entailment="ENTAILMENT" id="1" task="IR">
    <h>The organization: Develops and implements anti-counterfeit policy and procedures that include the means to detect and prevent counterfeit components from entering the information system; and
      Reports counterfeit information system components to [Selection (one or more): source of counterfeit component; [Assignment: organization-defined external reporting organizations];
      [Assignment: organization-defined personnel or roles]].</h>
    <t>The organization protects against supply chain threats to the information system, system component, or information system service by employing [Assignment: organization-defined security
      safeguards] as part of a comprehensive, defense-in-breadth information security strategy.</t>
  </pair>
  <pair entailment="ENTAILMENT" id="2" task="IR">
    <h>The organization: Develops, approves, and maintains a list of individuals with authorized access to the facility where the information system resides; Issues authorization credentials for
      facility access; Reviews the access list detailing authorized facility access by individuals [Assignment: organization-defined frequency]; and Removes individuals from the facility access
      list when access is no longer required.</h>
    <t>The organization implements an insider threat program that includes a cross-discipline insider threat incident handling team.</t>
  </pair>
  <pair entailment="ENTAILMENT" id="3" task="IR">
    <h>The organization: Authorizes connections from the information system to other information systems through the use of Interconnection Security Agreements; Documents, for each interconnection,
      the interface characteristics, security requirements, and the nature of the information communicated; and Reviews and updates Interconnection Security Agreements [Assignment:
      organization-defined frequency].</h>
    <t>The organization: Requires that providers of external information system services comply with organizational information security requirements and employ [Assignment: organization-defined
      security controls] in accordance with applicable federal laws, Executive Orders, directives, policies, regulations, standards, and guidance; Defines and documents government oversight and
      user roles and responsibilities with regard  to external information system services; and Employs [Assignment: organization-defined processes, methods, and techniques] to monitor security
      control compliance by external service providers on an ongoing basis.</t>
  </pair>
  <pair entailment="ENTAILMENT" id="4" task="IR">
    <h>A set of policies for information security shall be defined, approved by management, published and communicated to employees and relevant external parties.</h>
    <t>The organization: Develops, documents, and disseminates to [Assignment: organization-defined personnel or roles]: Reviews and updates the current: An incident response policy that addresses
      purpose, scope, roles, responsibilities, management commitment, coordination among organizational entities, and compliance; and Procedures to facilitate the implementation of the incident
      response policy and associated incident response controls; and Incident response policy [Assignment: organization-defined frequency]; and Incident response procedures [Assignment:
      organization-defined frequency].</t>
  </pair>
  <pair entailment="ENTAILMENT" id="5" task="IR">
    <h>The organization: Configures the information system to provide only essential capabilities; and Prohibits or restricts the use of the following functions, ports, protocols, and/or services:
      [Assignment: organization-defined prohibited or restricted functions, ports, protocols, and/or services].</h>
    <t>The information system enforces approved authorizations for controlling the flow of information within the system and between interconnected systems based on [Assignment: organization-define
      information flow control policies].</t>
  </pair>
```

Figure 2: The image shows a small section of the xml.

We start by computing the Part-Of-Speech (POS) tags of the words. Those are necessary to extract the features in the last step. We used OpenNLP[10] to obtain the POS tags of each word. Then, we tokenized the sentences using the NLTK[11] module. We filtered the stopwords out using the list provided by this latter framework. We also used a regular expression to remove all non-alphanumerical tokens because they are not relevant to recognize the TE relation. Finally, we stemmed and lowercased the remaining tokens to obtain a less diversified vocabulary.

Once the list of salient tokens is obtained through the above mentioned steps, we proceeded to extract the features for the classifiers. We first selected the POS tags corresponding

[10] `https://opennlp.apache.org`

[11] `https://www.nltk.org`

to the remaining words; then, we computed words n-grams and POS n-grams with n comprises in the range $[1, 5]$. We used a Term Frequency - Inverse Document Frequency schema to weight the extracted n-grams and to obtain the features.

3.2 Ablation Study

In the previous section, we said that the classifiers will use both word n-grams and POS n-grams. In this section, we will analyze the impact of these features on the performances. For this evaluation, we will compare the *Support Vector Machine* (SVM) classifier with the *Maximum Entropy* (ME) of EOP framework since both generally perform well in RTE tasks. We will use a *Random* classifier, which assigns the entailment relation with a probability of 0.5, as a baseline one.

We will train and test the classifiers on the *cybersecurity entailment* dataset to check the impact of the features. Since we do not have a testset, we will perform a cross-fold evaluation, with the fold number sets to 10. Each fold contains about 175 TE pairs. In detail, we will train the classifier on 9 folds and test on the remaining one. We will repeat this process leaving out a different fold for the test. Each classifier will be trained using the following features:

unigram: The classifier uses only unigrams as features. We decided to use such features as a baseline;

n-grams: The classifier uses n-grams as features, with n comprises in the range $[1, 5]$;

n-grams + POS: The classifier uses both word n-grams and POS tag n-grams, with n comprises in the range $[1, 5]$.

Table 2 reports the result of this evaluation. We can see that the n-grams slightly increased the accuracy of both classifiers. The accuracy is further increased with the adoption of the POS n-grams. It is interesting to notice that the n-grams had a major impact on the ME classifier than on the SVM one.

Model		Accuracy
SVM	+ unigrams	82.12%
	+ n-grams	82.58%
	+ n-grams + POS	83.06%
ME	+ unigrams	82.04%
	+ n-grams	82.75%
	+ n-grams + POS	83.10%

Table 2: The table reports the accuracy of ME and SVM classifiers with the different features.

3.3 Evaluation

In this section, we will evaluate several classifiers to check which one performs better in recognizing the expressed TE relation. The proposed classifiers are:

Random: it assigns a label randomly to each pair in the *cybersecurity entailment* dataset, which has a fixed accuracy of 50%. We used this classifier as baseline;

SVM: a Support Vector Machine that uses the extracted features (word and POS n-grams) to classify the pairs;

NB: a Naive Bayes classifier that uses the same features of SVM;

RF: since each pair of premise and hypothesis could contain specific words or POS tags that bring the entailment or neutral relation out, we decided to use a Random Forest classifier to capture them. The Random Forest creates a decision tree in which each branch contains word and POS n-grams features useful to distinguish the relation;

ME: a Maximum Entropy classifier with word and POS n-grams features. We used the implementation provided by the EOP framework since it contains state-of-the-art methods and classifiers for the TE task;

ME+WN+VO: it extends the features of the previous Maximum Entropy classifier with Wordnet [18] synsets (WN) and Verb Ocean [10] (VO) classes, i.e. a semantic network for verbs. VO reports for each verb: (1) the semantic relation with other verbs (e.g., to make and to create have a similarity relation), (2) if the verb is transitive and (3) if it is symmetric. Those features are used to handle possible periphrases and synonyms in the pairs. As for *ME*, we used the implementation provided by the EOP framework.

For the SVM, Naive Bayes and Random Forest classifiers, we used their implementation provided by the scikit-learn framework[12].

For the first evaluation, we decided to check if it is possible to train a classifier on the *cybersecurity entailment* dataset and generalize on similar data. Since we do not have a testset, we used the cross-fold validation. We divided the dataset in 10-fold, training the classifiers on nine folds and testing on the remaining one. Table 3 reports the results of this evaluation.

From the table, we can see that the SVM, the ME and the ME+WN+VO classifiers are able to recognize the relations expressed in those pairs, obtaining outstanding results. We can also notice that both the ME classifiers obtained an accuracy higher than the SVM, about 0.04 percentage points; since such difference is not significant, it is possible to use either the SVM or the ME. Both ME and ME+WN+VO classifiers have the same accuracy, meaning that the addition of WordNet synsets and Verb Ocean classes to the features is not relevant to recognize the TE relation.

We analyzed the errors made by SVM and ME classifiers. We found that they tend to mistake a neutral relation for an entailment one when both the premise and the hypothesis regard different topics of the same argument (e.g., auditing records storage vs. auditing events). An example of missclassified pair follows:

[12] `https://scikit-learn.org/stable/`

Classifier	Avg. Accuracy
Random	50%
SVM	83.06%
NB	82.90%
RF	79.42%
ME	83.10%
ME+WN+VO	83.10%

Table 3: The table reports the average accuracy for the cross-fold evaluation.

NIST: "*The organization: Schedules, performs, documents, and reviews records of maintenance and repairs on information system components in accordance with manufacturer or vendor specifications and/or organizational requirements; Approves and monitors all maintenance activities, whether performed on site or remotely and whether the equipment is serviced on site or removed to another location; Requires that [Assignment: organization-defined personnel or roles] explicitly approve the removal of the information system or system components from organizational facilities for off-site maintenance or repairs; Sanitizes equipment to remove all information from associated media prior to removal from organizational facilities for off-site maintenance or repairs; Checks all potentially impacted security controls to verify that the controls are still functioning properly following maintenance or repair actions; and Includes [Assignment: organization-defined maintenance-related information] in organizational maintenance records.* "

ISO: "*The organization: Documents and monitors individual information system security training activities including basic security awareness training and specific information system security training; and Retains individual training records for [Assignment: organization-defined time period].* "

In the proposed example, both the controls regard the *information system*, but they are not related to the same topic. The NIST one describes that the organization should maintain documents regarding maintenance activities and changes to the information systems, while the ISO one regards the training activities on the information systems.

We conducted a second evaluation to see whether it is possible to train the classifiers on a dataset, and use such acquired knowledge to recognize the relation expressed in our own one. In other words, we would like to verify if it is possible to generalize on our TE pairs. For this evaluation, we trained the classifiers using the legal textual entailment dataset proposed in COLIEE 2019[13] task 2. The dataset is composed of 362 pairs, divided into 182 pairs that express an entailment relation and 182 that express a contradiction one. Since this dataset does not present any neutral relation, we treated the neutral pairs of our dataset as negative ones to perform a proper evaluation. We applied the same preprocessing phase

[13] `https://sites.ualberta.ca/~rabelo/COLIEE2019/`

to the COLIEE trainset. Table 4 reports the results of this second evaluation. From the table, we can notice that only the SVM classifier slightly surpassed the Random one. Those results highlight the fact that our cybersecurity dataset contains complex pairs, making hard to generalize from a legal TE dataset to our one.

Classifier	Accuracy
Random	50%
SVM	50.48%
NB	49.44%
RF	48.86%
ME	50%
ME+WN+VO	50%

Table 4: The table reports the results obtained training the classifier on COLIEE dataset and testing on our one.

Finally, we conducted a third evaluation to see whether it is possible to transfer the knowledge that the classifier acquired from our dataset to other legal-based ones. For this experiment, we decided to evaluate the classifiers on the COLIEE testset. For a completed evaluation, we also reported the accuracy of the classifiers when they are trained and tested on only the COLIEE one. We expect that the accuracy will be high in this latter case, surpassing certainly the random classifier, while being lower for the generalization from the *cybersecurity entailment* dataset to the COLIEE one. Table 5 reports the results of this last evaluation.

Trainset	Classifier	Accuracy
-	Random	50%
COLIEE	SVM	71.11%
	NB	64.44%
	RF	67.80%
	ME	71.20%
	ME+WN+VO	71.06%
Cybersecurity	SVM	45.55%
	NB	46.70%
	RF	47.00%
	ME	47.00%
	ME+WN+VO	47.00%

Table 5: The table reports the results obtained training the classifier on COLIEE dataset and testing on our one.

As we expected, the accuracy of the classifiers trained and tested on COLIEE dataset surpassed the Random one. However, if we train them on our dataset, we obtain very poor performances; in this latter case, the classifiers have a lower accuracy, meaning that they found difficult to generalize on unseen data. Those results confirm again that our dataset contains more complex and semantic distant pairs than the COLIEE ones.

This could be verified computing the cosine distance between premise (or hypothesis) sentences of our dataset with the ones of COLIEE. More in detail, we calculated the average cosine distance between the premise (or hypothesis) sentences of the *cybersecurity entailment* dataset and the COLIEE ones. Table 6 reports the cosine distance for both the premise and hypothesis. From the table, it is possible to see that the two datasets do not have neither a jargon nor a syntactic structure in common.

Pairs	Cosine Distance
Premise	0.97
Hypothesis	0.95

Table 6: This table shows the cosine distance between the Cybersecurity dataset and the COLIEE one.

We report a distant pair for both the Premise and the Hypothesis. Table 7 shows a Premise pair and its score, while Table 8 shows an Hypothesis pair.

4 Related Works

Nowadays, Recognizing Textual Entailment (RTE) is an interesting task since it predicts the relation of two sentences. For instance, in legal domain could be used to see whether a law has a relation (i.e., entails) another one, or in case of European Union, we can see whether a law of a member state implements a directive of the EU (cf. [20]).

In general, research on generic RTE is conducted with the use of Neural Networks, where one important research works is [8]. In the article, the authors proposed both a dataset constructed through crowd-sourcing, and a Multi-Layer Perceptron (MLP) to classify the relation of a <premise, hypothesis> pair. After this article, researcher started to experiment with deep-learning models, also re-adapting idea coming from different NLP fields, such as Machine Translation. [26] proposed an encoder with attention for textual entailment. First, the authors sequentially read the premise and the hypothesis tokens with an LSTM, producing a list of encoded representation for the words. Then, they applied an attention mechanism to understand the correlation between the premise and the hypothesis words. They found that the attention mechanism is able to capture small semantic difference (e.g., the colour) in similar sentences. Finally, [29, 12, 24] found that, for some datasets, the hypothesis is all you need. According to them, the hypothesis contains very salient information that can be used by a Neural Network to unravel the relation. [23] used such models to evaluate Natural Language Inference Problems, defining several evaluation frameworks. Other works related to the legal domain include [6, 19, 2, 1].

Cybersecurity	The organization: Establishes and makes readily available to individuals requiring access to the information system, the rules that describe their responsibilities and expected behavior with regard to information and information system usage; Receives a signed acknowledgment from such individuals, indicating that they have read, understand, and agree to abide by the rules of behavior, before authorizing access to information and the information system; Reviews and updates the rules of behavior [Assignment: organization-defined frequency]; and Requires individuals who have signed a previous version of the rules of behavior to read and re-sign when the rules of behavior are revised/updated.
COLIEE	The proceeding at issue was a Motion for Summary Judgment under the previous Rules of the Court in regard to summary judgments. The Rules have been amended prior to the hearing of the motion to provide for summary trials but those amendments had no material effect on this matter.
Distance score	0.75

Table 7: The table reports a Premise sentence pair with its cosine distance score.

Cybersecurity	Information security shall be addressed in project management, regardless of the type of the project.
COLIEE	Further, this Court and the Federal Court of Appeal have both observed that a statement of claim should only be struck in the clearest and most obvious of cases.
Distance score	0.79

Table 8: The table reports an Hypothesis sentence pair with its cosine distance score.

Other researchers, instead, tried to apply the Recognizing Textual Entailment task on different domains, also starting competitions to see which models could perform well. In the field of Legal Informatics, we can find COLIEE (Competition on Legal Information

Extraction/Entailment)[14]. COLIEE started in 2014, and defined a competition every year up to now. Each competition is composed of four tasks: two regarding information retrieval on legal text, one regarding question answering and one regarding RTE on legal text. The task datasets are free to access upon request. In this paper, we decided to use their dataset for RTE in order to train our classifiers, since our dataset, to the best of our knowledge, is the first one regarding the cybersecurity domain. In this competition, [28] proposed a Multi-Layer Perceptron (MLP), composed of two hidden layers, with a decomposable attention model to find relations between words pairs. In detail, they started collecting articles from a civil code. Then, they ranked those articles according to a given query. Finally, they paired the best article (after the ranking) with the query to construct the training dataset for the MLP. [9] proposed a method similar to the one in [28], where they used n-grams, extracted using lexical and morphological characteristics, to retrieve articles from the civil code. Another one close to the work of Son et al. is [15]. In this work, the authors tried a convolutional neural network to see whether two legal articles are related to each other or not. Finally, [14] proposed a complex model to solve both legal information retrieval and textual entailment for COLIEE 2016. For the former one, they proposed an ensemble similarity method using least mean square and linear discriminant analysis. For the latter, they applied a majority vote schema of three classifiers: a decision tree, an SVM, and a convolutional neural network. As features for the classifiers, they used word overlap, cosine similarity, WordNet [18] similarity score, and substring similarity.

5 Conclusion

We presented a dataset for Recognizing Textual Entailment on the legal domain. All pairs of the dataset regard cybersecurity controls extracted from NIST, ISO and ISO/IEC 27001 documents. To the best of our knowledge, this is the first dataset for cybersecurity RTE.

We conducted three evaluations on our dataset using several classifiers. We first checked whether it is possible to train a classifier to recognize the relations expressed in our dataset. Since there is no testset, we used a cross-fold evaluation. We obtained an average accuracy of about 83% for the Support Vector Machine and the Maximum Entropy classifiers. We also reported that only word and POS tag n-grams are relevant as features to predict the relation. This is also confirmed by the ablation study. However, the classifiers tend to predict the wrong label when both the premise and hypothesis regard different aspects of the same topic (e.g., information system training vs information system maintenance). To solve this problem, we think that the classifiers require features that are able to capture the the topics of the NIST and ISO controls. For such reason, we will adopt the Topic Model proposed by [5].

We then performed a second evaluation, checking whether it is possible to transfer the knowledge acquired from a legal dataset for RTE to our one. Thus, we trained the classifiers using Task 2 dataset of COLIEE competition. We obtained an accuracy of about 50.48%, slightly surpassing the *Random* classifier. Such analysis showed that our dataset contains complex pairs, for both language and content, that do not allow classifiers trained

[14] `https://sites.ualberta.ca/~rabelo/COLIEE2019/`

on other datasets to generalize well. This has been confirmed by the third evaluation, where we evaluated if it is possible to transfer the knowledge acquired on our dataset to other legal ones for RTE. We decided to test the classifiers on the COLIEE testset, obtaining an accuracy of about 47%.

Finally, we discovered that all the proposed models have an accuracy on entailment pairs close to 100%. They however find difficult to recognize neutral pairs, obtaining an accuracy on these pairs at most of 10%. We think that a further classification of the pairs into the three classes *entailment*, *neutral* and *contradiction* will be useful to boost the performance of the machine learning models.

In our future works, we aim at integrating and inter-linking more cybersecurity standards. Specifically, we want to create a *unified* inter-connected corpus of technical documents in Natural Language for the cybersecurity domain, on which training and evaluating RTE classifiers to be later used by auditors as well as by companies collaborating with them, such as Nomotika SRL.

Acknowledgments

This research was supported by the EU's Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No 690974 for the project "MIREL: MIning and REasoning with Legal texts" (`http://www.mirelproject.eu`). We would like to thank prof. Cleo Condoravdi for fruitful discussions and feedback during the secondment of Giovanni Siragusa and Livio Robaldo at Stanford University in the context of the MIREL project.

References

[1] Kolawole Adebayo, Luigi Di Caro, and Guido Boella. Siamese network with soft attention for semantic text understanding. In *Proceedings of the 13th International Conference on Semantic Systems*, pages 160–167, 2017.

[2] Kolawole John Adebayo, Luigi Di Caro, Livio Robaldo, and Guido Boella. Textual inference with tree-structured lstm. In *BNCAI*, pages 17–31, 2016.

[3] G. Ajani, G. Boella, L. DI Caro, L. Robaldo, L. Humphreys, S. Praduroux, P. Rossi, and A. Violato. The European Taxonomy Syllabus: A multi-lingual, multi-level ontology framework to untangle the web of European legal terminology. *Applied Ontology*, 11(4), 2016.

[4] Luisa Bentivogli, Ido Dagan, and Bernardo Magnini. *The Recognizing Textual Entailment Challenges: Datasets and Methodologies*, pages 1119–1147. Springer Netherlands, Dordrecht, 2017.

[5] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[6] G. Boella, L. di Caro, L. Humphreys, L. Robaldo, and L. van der Torre. NLP Challenges for Eunomos, a Tool to Build and Manage Legal Knowledge. Proceedings of the International Conference on Language Resources and Evaluation, 2012.

[7] G. Boella, G. Governatori, A. Rotolo, and L. van der Torre. A logical understanding of legal interpretation. In *Int. Conference on the Principles of Knowledge Representation and Reasoning*, 2010.

[8] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[9] Danilo S Carvalho, Minh-Tien Nguyen, Chien-Xuan Tran, and Minh-Le Nguyen. Lexical-morphological modeling for legal text analysis. In *JSAI International Symposium on Artificial Intelligence*, pages 295–311. Springer, 2015.

[10] Timothy Chklovski and Patrick Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004.

[11] D. Grossi., C. Meyer, and F. Dignum. The many faces of counts-as: A formal analysis of constitutive-rules. *Journal of Applied Logic*, 6(2), 2008.

[12] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 107–112, 2018.

[13] H. Hart. *The Concept of Law*. Oxford: Clarendon Press, 1994.

[14] Kiyoun Kim, Seongwan Heo, Sungchul Jung, Kihyun Hong, and Young-Yik Rhim. An ensemble based legal information retrieval and entailment system. In *Tenth International Workshop on Juris-informatics (JURISIN)*, 2016.

[15] Mi-Young Kim, Randy Goebel, and S Ken. Coliee-2015: evaluation of legal question answering. In *Ninth International Workshop on Juris-informatics (JURISIN 2015)*, 2015.

[16] Daniel Z. Korman, Eric Mack, Jacob Jett, and Allen H. Renear. Defining textual entailment. *Journal of the Association for Information Science and Technology*, 69(6), 2018.

[17] Bernardo Magnini, Roberto Zanoli, Ido Dagan, Kathrin Eichler, Günter Neumann, Tae-Gil Noh, Sebastian Pado, Asher Stern, and Omer Levy. The excitement open platform for textual inferences. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 43–48, 2014.

[18] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[19] Rohan Nanda, Kolawole John Adebayo, Luigi Di Caro, Guido Boella, and Livio Robaldo. Legal information retrieval using topic clustering and neural networks. In *COLIEE@ ICAIL*, pages 68–78, 2017.

[20] 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 *Proceedings of the 16th Edition of the International Conference on Articial Intelligence and Law*, ICAIL '17, pages 149–158, New York, NY, USA, 2017. ACM.

[21] Sebastian Padó, Tae-Gil Noh, Asher Stern, Rui Wang, and Roberto Zanoli. Design and realization of a modular architecture for textual entailment. *Natural Language Engineering*, 21(2):167–200, 2015.

[22] Monica Palmirani, Michele Martoni, Arianna Rossi, Cesare Bartolini, and Livio Robaldo. Legal ontology for modelling GDPR concepts and norms. In Monica Palmirani, editor, *Legal Knowledge and Information Systems - JURIX 2018: The Thirty-first Annual Conference, Groningen, The Netherlands.*, pages 91–100, 2018.

[23] Adam Poliak, Aparajita Haldar, Rachel Rudinger, J Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 67–81, 2018.

[24] Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, 2018.

[25] 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. *Journal of Logic, Language and Information*, to appear, 2019.

[26] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. Reasoning about entailment with neural attention. In *Proceedings of the 2015 International Conference on Learning Representations*, 2015.

[27] A. Rotolo and C. Roversi. Constitutive rules and coherence in legal argumentation: The case of extensive and restrictive interpretation. *Legal Argumentation Theory*, 2012.

[28] Nguyen Truong Son, Viet-Anh Phan, and Nguyen Le Minh. Recognizing entailments in legal texts using sentence encoding-based and decomposable attention models. In *COLIEE@ ICAIL*, pages 31–42, 2017.

[29] Masatoshi Tsuchiya. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, 2018.

 Received 1 February 2021

Time, Defeasible Logic and Belief Revision: Pathways to Legal Dynamics

Luciano H. Tamargo and Diego C. Martinez
ICIC–UNS–CONICET, Universidad Nacional del Sur, Argentina
{lt,dcm}@kcs.uns.edu.ar

Antonino Rotolo
CIRSFID, University of Bologna, Italy
antonino.rotolo@unibo.it

Guido Governatori
Data61, CSIRO, Australia
guido.governatori@data61.csiro.au

Abstract

In order to properly model norm change in the law, temporal aspects of legal dynamics must be considered. Since there exist several time-based features of law that should be studied, we discuss two interesting approaches: one based on defeasible logic and the other based on belief revision. Each of these makes use of one of the two classic forms of reasoning about time: point-based and interval-based. Both formalisms provide the necessary logical infrastructure to address the characterization of complex behaviour of legal dynamics.

1 Introduction and Background

One peculiar feature of many normative systems, such as the law, is that it necessarily takes the form of a dynamic normative system [24, 23]. Despite the importance of norm-change mechanisms, the logical investigation of legal dynamics is still relatively underdeveloped. However, recent contributions exist and this section is devoted to a brief sketch of this rapidly evolving literature.

Alchourrón and Makinson were the first to logically study the changes of a *legal code* [2, 3, 1]. The addition of a new norm n causes an enlargement of the code,

consisting of the new norm plus all the regulations that can be derived from n. Alchourrón and Makinson distinguish two other types of change. When the new norm is incoherent with the existing ones, we have an *amendment* of the code: in order to coherently add the new regulation, we need to reject those norms that conflict with n. Finally, *derogation* is the elimination of a norm n together with whatever part of the legal code that implies n. Alchourrón, Gärdenfors and Makinson [4] inspired by the works above proposed the so called general AGM framework for belief revision. This area has been proved to be a very fertile one and the phenomenon of revision of logical theories has been thoroughly investigated. As is well-known, the AGM framework distinguishes three types of change operation over theories. Contraction is an operation that removes a specified sentence ϕ from a given theory Γ (a logically closed set of sentences) in such a way as Γ is set aside in favour of another theory Γ^-_ϕ which is a subset of Γ not containing ϕ. Expansion operation adds a given sentence ϕ to Γ so that the resulting theory Γ^+_ϕ is the smallest logically closed set that contains both Γ and ϕ. Revision operation adds ϕ to Γ but it is ensured that the resulting theory Γ^*_ϕ be consistent [4]. Alchourrón, Gärdenfors and Makinson argued that, when Γ is a code of legal norms, contraction corresponds to norm derogation (norm removal) and revision to norm amendment.

It is then natural to ask if belief revision offers a satisfactory framework for the problem of norm revision in the law. Some of the AGM axioms seem to be rational requirements in a legal context, whereas they have been criticised when imposed on belief change operators. An example is the *success* postulate, requiring that a new input must always be accepted in the belief set. It is reasonable to impose such a requirement when we wish to enforce a new norm or obligation. However, it gives rise to irrational behaviours when imposed to a belief set, as observed in [14].

The AGM operation of contraction is perhaps the most controversial one, due to some postulates such as recovery [16, 32], and to elusive nature of legal changes such as derogations and repeals, which are all meant to contract legal effects but in remarkably different ways [16]. Standard AGM framework is of little help here: it has the advantage of being very abstract—it works with theories consisting of simple logical assertions—but precisely for this reason it is more suitable to capture the dynamics of obligations and permissions rather than the one of legal norms.

Difficulties behind AGM have been considered and some research has been carried out to reframe AGM ideas within reasonably richer rule-based logical systems able to capture the distinction between norms and legal effects [28, 26]. However, these attempts suffer from some drawbacks: they fail to handle reasoning on deontic effects and are based on a very simple representation of legal systems.

In fact, it is hard in AGM to represent how the same set of legal effects can be

contracted in many different ways, depending on how norms are changed. These difficulties have been addressed in logical frameworks combining AGM ideas with richer rule-based logical systems, such as standard or Defeasible Logic [26, 18] or Input/Output Logic [8, 9, 28]. [32] suggested a different route, i.e., employing in the law existing techniques—such as iterated belief change, two-dimensional belief change, belief bases, and weakened contraction—that can obviate problems identified in [16] for standard AGM.

In general, any comprehensive logical model of norm change in the law has to take care of the following aspects:

1. the law usually regulates its own changes by setting specific norms whose peculiar objective is to change the system by stating what and how other existing norms should be modified; for instance, in most countries the Constitution states that only the Congress have powers to lay and regulate taxes. Even more, the Constitution states, by a norm, how to change or amend its own body of norms.

2. since legal modifications are derived from these peculiar norms, they can be in conflict and so are defeasible; for instance, some US states requires non-english foreign driver licenses to be accompanied by the International Drivers Permit. However, in 1989 US and Canada agreed to recognize each other's licenses, even french-written licenses. Hence, norms are contradictory regarding the documentation that a french-canadian driver must show to authorities.

3. legal norms are qualified by temporal properties, such as the time when the norm comes into existence and belongs to the legal system, the time when the norm is in force, the time when the norm produces legal effects, and the time when the normative effects hold. For instance, Belarus established that several laws passed before 1996 ceased to be enforced in the exact moment the President issues the new Constitution. In the United States, the 18th Amendment prohibiting the manufacture of liquor was passed in 1919 and repealed later in 1933. The end of this prohibition was established in turn by another Amendment (the 21st) that also establishes that this amendment "*shall be inoperative unless it shall have been ratified (...) within seven years from the date of the submission*".

To sum up, AGM-like frameworks have the advantage of being very abstract but work with theories consisting of simple logical assertions. For this reason, it is perhaps suitable to capture the dynamics of obligations and permissions, not of norms: the former ones are just possible effects of the application of norms and

their dynamics do not necessarily require to remove or revise norms, but correspond in most cases to instances of the notion of *norm defeasibility* [16].

Addressing the above aspects has triggered new research lines in recent years, which break down in the following two approaches:

- Normative dynamics can be modelled by combining logical systems for temporal and defeasible reasoning: previous works [15, 19, 16] have proposed to combine Defeasible Logic with some basic forms of temporal logics;
- Another route is rather to enrich belief revision techniques by adding several temporal dimensions: this has been done in works such as [29, 30].

The layout of the article is as follows. Section 2 introduces the importance of time in legal norms and shows an example to motivate some ideas in the area of legal dynamics. Section 3 summarizes the first approach mentioned above, in which it is described how the Defeasible Logic was extended with temporal parameters to allow for reasoning about the times specified inside norms, and it is described how consider a legal system as a time-series of its versions, where each version is obtained from previous versions by some norm changes. Section 4 summarizes the second approach mentioned above, in which it is proposed a belief revision operator that considers time interval in the revision process. Finally, in Section 5 conclusions are offered and ideas for future work are given.

2 Preamble: Why Does Time Matter?

Legal norms are qualified by temporal properties, such as the time when the norm comes into existence and belongs to the legal system or the time when the norm is in force. Suppose that a municipality establishes that all taxis licensed since 2015 must be all-yellow, and a couple of years later the city adds a new rule establishing that all taxis with license starting in 2018 must be all-black. Hence, the yellow-taxi rule only applies for passenger cars with a valid license from 2015 to 2017. However, this is true only years later, after the introduction of the black-taxi rule.

Since all these properties can be relevant when legal systems change, [17] argued that failing to consider the temporal aspects of legal dynamics poses a serious limit to correctly model norm change in the law.

2.1 The Problem and a Motivating Example

As we have briefly mentioned above, belief revision, and specifically the AGM paradigm, has been advocated to be an elegant and abstract model for legal change.

Its has been, however, argued that standard belief techniques do not capture the following aspects of the law [17]:

1. the law regulates its own changes by issuing norms stating what and how other norms should be modified;
2. legal modifications can be in conflict and so are defeasible;
3. legal norms are qualified by temporal properties, e.g. the time when the norm is in force.

The general temporal model, as proposed in [17] assumes that all legal norms are qualified by different temporal parameters:

- the time when the norm comes into existence and belongs to the legal system,
- the time when the norm is in force,
- the time when the norm produces legal effects (it is applicable), and
- the time when the normative effects (conclusions) hold.

Indeed, it is common legislative practice that, once a legal provision is enacted (for example, the Italian 2018 budget law was enacted on 23 December 2017), its force can for instance be postponed to a subsequent time (for example, the Italian 2018 budget law was in force since 1 January 2018). Similarly, a part of a certain provision, which is in force since a certain time t, can be effective (i.e., can be applied) since a different time t' (for example, the Italian 2018 budget law, which was in force since 1 January 2018, at art. 1, par. 253 states that par. 252 will be applicable since 1 January 2019), or any provision can produce effects that hold retroactively (for example, art. 1 of Italian 2018 budget law, par. 629, states that certain tax effects cover cases since December 2017).

In [30], for example, the authors concentrate on issue 3 in the list above, i.e., how to integrate belief revision with time in the law. As regards issue 2, in that article, the authors do not work directly on rule-based defeasible reasoning, but they define a revision operator that may remove rules when needed or adapt intervals of time when contradictory norms are introduced in the system: for instance, if n is effective from 2001 to 2008 and a contradictory norm n' is added at 2006, we know that n is still effective from 2001 to 2005.

Let us now present a concrete example that serves to motivate the main ideas proposed in [30], an approach that we will recall in Section 4. It involves information and rules referring to intervals of time in which some taxes applies.

Example 1. *Consider the following pieces of information regarding a legislative attempt to ease tax pressure for people that have been unemployed.*

(a) A citizen was unemployed from 1980 to 1985.

(b) If unemployed from 1980 to 1983, then a tax exemption applies from 1984 to 1986, in order to increase individual savings.

(c) New authorities in government revoke tax exemption for years 1985 and 1986.

(d) Tax exemption reinstated for the year 1985 due to agreements with labor unions. ∎

However, later on the legislator approved a new provision establishing that finally there is no tax-exemption for all citizens for the years 1985 and 1986.

Here some rules are produced and, as it happens in legislative bodies, norms change later according to the political and economical context. Rule (a) provides time-bounded information: only between 1980 and 1985 the status of being unemployed holds for a given citizen. Rule (b) states that if some property (unemployed) holds between 1980 and 1983, then other property (tax exemption) holds between 1984 and 1986. Rule (c) establishes that this is no longer valid for a certain interval of time. This means that, from now on, rule (b) of tax exemption should not be applied in its original text. In other words, the intervals of rule (b) are *revised* according to new political positions. Finally, rules are revised again as a consequence of labor unions, only to be revoked later. In this example the general rule of tax-exemption is revised several times. This revision is actually about the moments in which this benefit can be applied. In fact, rule (c) solely demands a revision of the interval for tax exemption. Hence, it cannot be the case that there is a rule in the normative system that entails a tax exemption for 1985 and 1986. From (c) and (b), it can be concluded that the benefit is only applied to 1984. Therefore, (b) should be not used anymore and a new rule for 1984 should be introduced.

3 Defeasible Logic with Time for Modelling Legal Dynamics

Before illustrating in Section 4 how belief revision can be integrated by temporal reasoning, we recall in this section the other approach that we mentioned in the introduction.

In [15, 19, 16] Defeasible Logic was extended with temporal parameters. In particular the authors *temporalised* propositional Defeasible Logic. This means that

a temporal parameter is attached to the atomic elements of the logic, i.e., to the atomic propositions. For the logic it is assumed a discrete totally ordered set of instants of time $\mathcal{T} = \{t_0, t_1, t_2, \dots\}$. Based on this we can introduce the notion of *temporalised literals*. Thus if l is a plain literal, i.e., $l \in \text{PlainLit}$, and $t \in \mathcal{T}$ then l^t is a temporalised literals. The intuitive interpretation of l^t is that l is true (or holds) at time t. Lit denotes the set of temporalised literals. Finally, given a time instant t and $y \in \{pers, tran\}$ we call the combination of (t, y) *duration specification*, and literals labelled with a duration specification are called *duration literals*. The labels *pers* and *tran* denotes the quality of being *transient* or *persistent*. A duration literal has the form $l^{(t,y)}$. We denote the set of duration literals DurLit. The reasoning mechanism occurs on a set of *rules*, which are supposed to represent legal rules. The signature of rules is

$$\text{Rule}\colon 2^{\text{Lit}} \times \text{DurLit} \tag{1}$$

this means that a rule has the following form

$$r\colon a_1^{t_1}, \dots, a_n^{t_n} \hookrightarrow c^{(t,y)} \tag{2}$$

where $y \in \{tran, pers\}$ and hence the conclusion of the rule may be *transient* or *persistent*.

The idea behind the distinction between a transient and persistent conclusion is whether the conclusion is guaranteed to hold for a single instant or it continues to hold until it is terminated. This is particular relevant for legal rules, since their conclusions are for example obligations (or, in general deontic effects), and obligations, once triggered, remain in force until they are complied with, violated, or explicitly terminated. Accordingly we can use the duration specification $(t, tran)$ to indicate that an obligation is in force at a specific time t, and must be fulfilled at that time, while the duration specification $(t, pers)$ establishes that a legal effect enters in force at time t.

The inference mechanism extends that of Defeasible Logic taking into account the temporal and durations specification. As in article [6], we equate arguments with rules, thus this is the same as saying that there is a (defeasible) rule such that all the elements in its antecedent are provable and the conclusion is $p^{(t',y)}$. To assert that p holds at time t we have the following steps:

1. Give an argument for p at time t';
2. Evaluate all counterarguments against it. Here, we have a few cases:
   (a) If the duration specification of p is $(t, tran)$ $(t' = t)$, then, the counterargument must be for the same time t given that p is ensured to hold only for t.

(b) If the duration specification of p is $(t', pers)$, then t' can precede t and we can 'carry' over the conclusion from previous times. In this case, the counterarguments we have to consider are all rules whose conclusion has a duration specification (t'', z) such that $t' \leq t'' \leq t$.

3. Rebut the counterarguments. This is the same as the corresponding step of basic defeasible logic, the only thing to pay attention to is that when we rebut with a stronger argument, the stronger argument should have t'' in the duration specification of the conclusion.

The general idea of the conditions outline above is that it is possible to assert that something holds at time t, because it did hold at time t', $t' < t$, by persistence, but there must be no reasons to terminate it. Thus new information defeats previous one.

3.1 From Rules to Meta-Rules

The temporal Defeasible Logic just presented allows for reasoning about the times specified inside norms, but it is not able to capture the natural evolution of legal systems, where new norms are issued, and existing norms are revised or derogated. To obviate this problem [16] proposes to consider a legal system as a time-series of its versions, where each version is obtained from previous versions by some norm changes, e.g., norms entering in the legal system, modification of existing norms, repeals of existing norms, ... This means that we can represent a legal system LS as a sequence

$$LS(t_1), LS(t_2), \ldots, LS(t_j) \tag{3}$$

where each $LS(t_i)$ is the snapshot of the rules (norms) in the legal system at time t_i. Graphically it can be represented by the picture in Figure 1.

A *rule* is a relation between a set of premises (conditions of applicability of the rule) and a conclusion. The admissible conclusions are either literals or rules themselves; in addition the conclusions and the premises will be qualified with the time when they hold. Two classes of rules can be considered: *meta-rules* and *proper rules*. Meta-rules describe the inference mechanism of the institution on which norms are formalised and can be used to establish conditions for the creation and modification of other rules or norms, while proper rules correspond to norms in a normative system. In what follows we will use Rule to denote the set of rules, and MetaRules for the set of meta-rules, i.e., rules whose consequent is a rule.

A *temporalised rule* is either an expression $(r\colon \bot)^{(t,x)}$ (the void rule) or $(r\colon \emptyset)^{(t,x)}$ (the empty rule) or $(r\colon A \hookrightarrow B)^{(t,x)}$, where r is a rule label, A is a (possibly empty) set of temporalised literals, B is a duration literal, $t \in \mathcal{T}$ and $x \in \{tran, pers\}$.

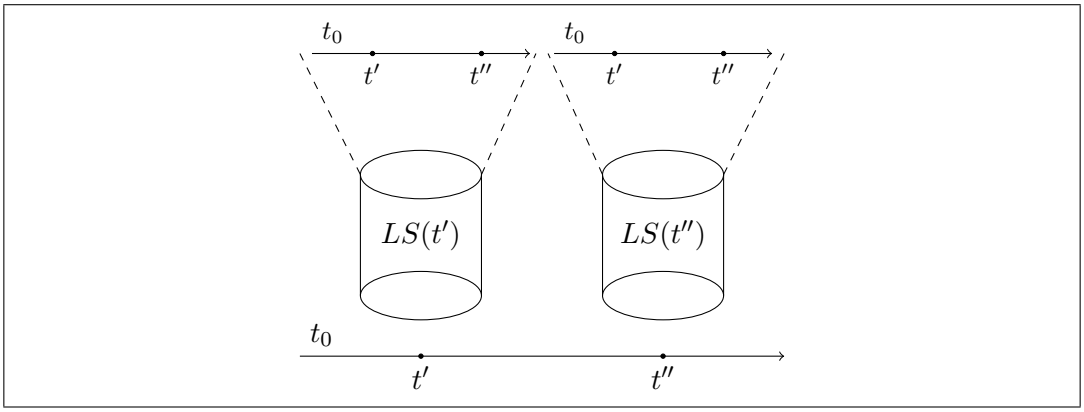


Figure 1: Legal System at t' and t''

We have to consider two temporal dimensions for norms in a normative system. The first dimension is when the norm is in force in a normative system, and the second is when the norm exists in the normative system from a certain viewpoint. So far temporalised rules capture only one dimension, the time of force. To cover the other dimension we introduce the notion of temporalised rule with viewpoint. A *temporalised rule with viewpoint* is an expression

$$(r\colon A \hookrightarrow B)^{(t,x)}@(t', y), \tag{4}$$

where $(r\colon A \hookrightarrow B)^{(t,x)}$ is a temporalised rule, $t' \in \mathcal{T}$ and $y \in \{tran, pers\}$.

Finally, meta-rules are introduced, that is, rules where the conclusion is not a simple duration literal but a temporalised rule. Thus a *meta-rule* is an expression

$$(s\colon A \hookrightarrow (r\colon B \hookrightarrow C)^{(t',x)})@(t, y), \tag{5}$$

where $(r\colon B \hookrightarrow C)^{(t',x)}$ is a temporalised rule, $r \neq s$, $t \in \mathcal{T}$ and $y \in \{tran, pers\}$. Notice that meta-rules carry only the viewpoint time (the validity time) but not the "in force" time. The intuition behind this is that meta-rules yield the conditions to modify a legal system. Thus they specify what rules (norms) are in a normative system, at what time the rules are valid, and the content of the rules. Accordingly, these rules must have an indication when they have been inserted in a normative system, but then they are universal (i.e., apply to all instants) within a particular instance of a normative system.

Every temporalised rule is identified by its rule label and its time. Formally we can express this relationship by establishing that every rule label r is a function

$$r\colon \mathcal{T} \mapsto \text{Rule}. \tag{6}$$

Thus a temporalised rule r^t returns the value/content of the rule 'r' at time t. This construction allows us to uniquely identify rules by their labels[1], and to replace rules by their labels when rules occur inside other rules. In addition there is no risk that a rule includes its label in itself. In the same way a temporalised rule is a function from $\mathcal{T}$ to Rule, we will understand a temporalised rule with viewpoint as a function with the following signature:

$$\mathcal{T} \mapsto (\mathcal{T} \mapsto \text{Rule}). \tag{7}$$

As we have seen above a legal system LS is a sequence of versions $LS(t_0), LS(t_1), \ldots$. The temporal dimension of *viewpoint* corresponds to a version of the legal system, while the temporal dimension of a *temporalised rule* corresponds to the time-line inside a version. Thus the meaning of an expression $r^{t_v}@t_r$ is that we take the value of the temporalised rule r^{t_v} in $LS(t_r)$. Accordingly, a version of LS is just a repository (set) of norms (implemented as temporal functions).

Accordingly, given a rule r, the expression $r^t@t'$ gives the value of the rule (set of premises and conclusion of the rule) at time t in the repository t'. The content of a void rule, e.g., $(r\colon \bot)^t@t'$ is $\bot$, while for the empty rule the value is the empty set. This means that the void rule has a value for the combination of the temporal parameters, while for the empty rule, the content of the rule does not exist for the given temporal parameters. Another way to look at the difference between the empty rule and the void rule is to consider that a rule is a relationship between a set of premises and a conclusion. For the void rule this relationship is between the empty set of premises and the empty conclusion; thus the rule exists but it does not produce any conclusion. For the empty rule, the relationship is empty, thus there is no rule. Alternatively, we can think of the function corresponding to temporalised rules as a partial function, and the empty rule identifies instants when the rule is not defined.

For a transient fully temporalised literal $l^{(t,x)}@(t', tran)$ the reading is that the validity of l at t is specific to the legal system corresponding to repository associated to t', while $l^{(t,x)}@(t', pers)$ indicates that the validity of l at t is preserved when we move to legal systems after the legal system identified by t'. An expression $r^{(t,tran)}$ sets the value of r at time t and just at that time, while $r^{(t,pers)}$ sets the values of r to a particular instance for all times after t (t included).

We will often identify rules with their labels, and, when unnecessary, we will drop the labels of rules inside meta-rules. Similarly, to simplify the presentation and when possible, we will only include the specification whether an element is persistent or transient only for the elements for which it is relevant for the discussion at hand.

[1]We do not need to impose that the function is an injective: while each label should have only one content at any given time, we may have that different labels (rules) have the same content.

Meta-rules describe the inference mechanism of the institution on which norms are formalised and can be used to establish conditions for the creation and modification of other rules or norms, while proper rules correspond to norms in a normative system. Thus a temporalised rule r^t gives the 'content' of the rule 'r' at time t; in legal terms it tells us that norm r is in force at time t. The expression

$$(p^{t_p}, q^{t_q} \Rightarrow (p^{t_p} \Rightarrow s^{(t_s,pers)})^{(t_r,pers)})@(t, tran) \tag{8}$$

means that, for the repository at t, if p is true at time t_p and q at time t_q, then $p^{t_p} \Rightarrow s^{(t_s,pers)}$ is in force from time t_r onwards.

A legal system is represented by a temporalised defeasible theory, called *normative theory*, i.e., a structure

$$(F, R, R^{\text{meta}}, \prec) \tag{9}$$

where F is a finite set of facts (i.e., fully temporalised literals), R is a finite set of rules, R^{meta} is a finite set of meta rules, and $\prec$, the superiority relation over rules is formally defined as $\mathcal{T} \mapsto (\mathcal{T} \mapsto \text{Rule} \times \text{Rule})$ accounting that we can have different instances of the superiority relation depending on the legal systems (external time) and the time when the rules involved in the superiority are evaluated[2].

The inference mechanism with meta-rules is essentially an extension of that of temporal defeasible logic, but it involves more steps. Rules are no longer just given, but they can be derived from meta-rules. Thus, to prove a conclusion x the first thing to do is to see if it is possible to derive a rule r supporting x. But we have to derive such rule at the appropriate time. Here, we want to remember that a rule is a function from time (validity time or version of a legal system) to time (when a rule is in force in a version of a legal system) to the content of the rule (relationship between a set of premises and a conclusion). The basic intuition is that a rule corresponds to a norm, and there could be several modifications of a norm, thus deriving a rule means to derive one of such modifications. As we shall see in the next section a meta-rule (or more generally a set of meta-rules) can be used to encode a modification of a norm. In general it is possible to have multiple (conflicting) modifications of a norm. Accordingly, to derive a rule, we have to check that there are no conflicting modifications[3] or the conflicting modifications are weaker than the current modification. The final consideration is that in this case we have two temporal dimensions, and the persistence applies to both.

[2]For instance, if we have $s \prec^{2007}_{Monday} r$ and $r \prec^{2007}_{Tuesday} s$, it means that, according to the regulation in force in 2007, on Monday rule s is stronger than rule r, but on Tuesday r is stronger than s.

[3]Two meta-rules are conflicting, when the two meta-rules have the same rule as their head, but with a different content.

3.2 An Example: Modifications on Norm Validity and Existence – Annulment vs. Abrogation

The expression *repeal* is sometimes used to generically denote the operation of norm withdrawal. However, at least two forms of withdrawal are possible: annulment and abrogation.

An *annulment* makes the target norm invalid and removes it from the legal system. Its peculiar effect applies *ex tunc*: annulled norms are prevented to produce all their legal effects, independently of when they are obtained. Annulments typically operate when the grounds (another norm) for annulling are hierarchically higher in the legal system than the target norm which is annulled: consider when a legislative provision is annulled (typically by the Constitutional Court) because it violates the constitution.

An *abrogation* works differently; the main point is usually that abrogations operate *ex nunc* and so do not cancel the effects that were obtained from the target norm before the modification. If so, it seems that abrogations cannot operate retroactively. In fact, if a norm n_1 is abrogated in 2012, its effects are no longer obtained after then. But, if a case should be decided at time 2013 but the facts of the case are dated 2011, n_1, if applicable, will anyway produce its effects because the facts held in 2011, when n_1 was still in force (and abrogations are not retroactive). Accordingly, n_1 is still in the legal system, even though is no longer in force after 2012. Abrogations typically operate when the grounds (another norm) for abrogating is placed at the same level in the hierarchy of legal sources of the target norm which is abrogated: consider when a legislative provision is abrogated by a subsequent legislative act.

Consider this case:

Example 2 (Abrogation vs Annulment)**.** *[***Target of the modification***]*

> *Legislative Act n. 124, 23 July 2008*
> *Art. 1. With the exception of the cases mentioned under the Articles 90 and 96 of the Constitution, criminal proceedings against the President of the Republic, the President of the Senate, the President of the House of Representatives, and the Prime Minister, are suspended for the entire duration of tenure. [. . .]*

In case of abrogation, we could have that the legislator enacts the following provision:

> *[***Abrogation enacted and effective at 1 January 2011***] Legislative Act n. 124, 23 July 2008 is abrogated.*

In case of (judicial) annulment, we would rather have

> *[**Annulment enacted and effective at 1 January 2011**] On account of Art. 3 of the Constitution [...] the Constitutional Court hereby declares the constitutional illegitimacy of Art. 1 of the Act n. 124, 23 July 2008.*

As we have recalled, the difference between the two cases is that the annulment has retroactive effects. In particular, let us focus on the following provisions from the Italian penal code:

> Art. 157 Italian of Penal Code – Terms of statute-barred penal provisions.
> When the terms for statute-barred penal effects expire, the corresponding crime is canceled [...]
> Art. 158 Italian Penal Code – Effectiveness of the terms of statute-barred penal provisions
> The effectiveness of terms of statute-barred penal provisions begins starting from the time when the crime was committed.
> Art. 159 Italian of Penal Code – Suspension of time limits for statute-barred penal effects.
> The terms for statute-barred penal effects [...] are suspended whenever the criminal proceedings are suspended under any legislative provisions [...]

Consider a hypothetical case where the Italian Prime Minister is accused in 2007 of accepting bribes at the beginning of 2006. Clearly, if Legislative Act n. 124 is abrogated in 2011, since abrogation has no retroactive effects, art. 159 of Italian Penal Code applied from 2008 to 2011, and so the counting of terms has been suspended between these two years. Hence, from the perspective of 2011 (immediately after the abrogation) the relevant time passed is two years and six months (2006, 2007, and until July 2008). Instead, if the act is annulled in 2011, more time has passed from the perspective of 2011, because it is as if the Legislative Act n. 124 were never enacted: from 2006 until 2011.

As we can see, modeling retroactive legal modifications is far from obvious. The logical model proposed in [16] and recalled in Section 3 offers a solution. In the next section we will illustrate the intuition and apply to the above example of annulment and abrogation.

3.3 Intermezzo – Temporal Dynamics and Retroactivity

As we have previously argued, if $t_0, t_1, \ldots, t_j$ are points in time, the dynamics of a legal system LS can be captured by a time-series $LS(t_0), LS(t_1), \ldots, LS(t_j)$ of

its versions. Each version of LS is like a norm repository: the passage from one repository to another is effected by legal modifications or simply by temporal persistence. This model is suitable for modeling complex modifications such as retroactive changes, i.e., changes that affect the legal system with respect to legal effects which were also obtained before the legal change was done.

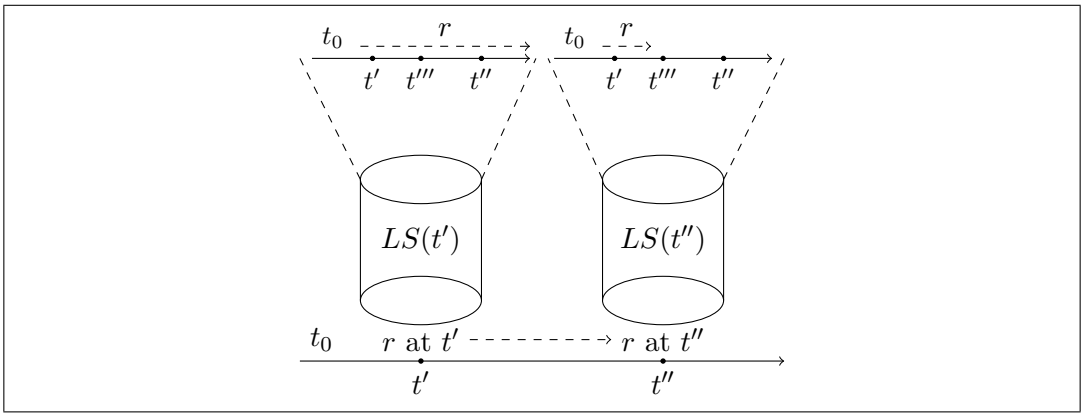


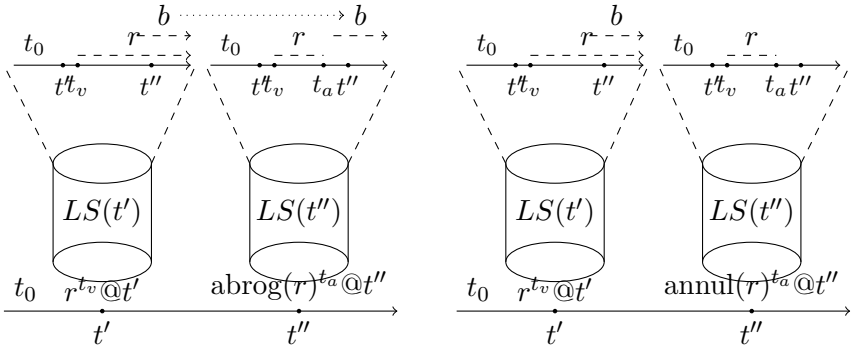
Figure 2: Legal System at t' and t''

The dynamics of norm change and retroactivity need to fully make use of the time-line within each version of LS (the time-line placed on top of each repository in Figure 2). Clearly, retroactivity does not imply that we can really change the past: this is "physically" impossible. Rather, we need to set a mechanism through which we are able to reason on the legal system from the viewpoint of its current version but *as if* it were revised in the past: when we change some $LS(i)$ retroactively, this does not mean that we modify some $LS(k)$, $k < i$, but that we move back from the perspective of $LS(i)$. Hence, we can "travel" to the past along this inner time-line, i.e., from the viewpoint of the current version of LS where we modify norms.

Figure 2 shows a case where the legal system LS and its norm r persist from time t' to time t'' and can have effects immediately from t'. Now, the figure represents the situation where r is retroactively repealed at t'' by stating that the modification applies from t''' (which is between t' and t'') onwards. The difference between abrogation and annulment is illustrated in Figures 3(a) and 3(b).

3.4 Modifications on Norm Validity and Existence: Annulment vs. Abrogation (Cont'd)

On account of our previous considerations, the cases in Example 2 can be reconstructed as follows.



(a) Abrogation. In $LS(t')$ rule r produces a persistent effect b. Literal b carries over by persistence to $LS(t'')$ even if r is no longer in force.

(b) Annulment. In $LS(t')$ rule r is applied and produces a persistent effect b. Since r is annulled in $LS(t'')$, b must be undone as well.

Figure 3: Abrogation and Annulment

Example 3 (Abrogation vs Annulment (cont'd))**.** *First of all, for the sake of simplicity let us*

- *only consider the case of Prime Minister (Legislative Act n. 124 mentions other institutional roles),*
- *assume that the dates of enactment and effectiveness coincide and are generically 2008,*
- *the duration of tenure covers a time span from 2008 to 2012,*

and formalize the corresponding fragment of art. 1 of Legislative Act n. 124 (23 July 2008) as follows:

$$\mathit{L.\ 124}: (\mathit{Crime}^{x}, \mathit{Tenure}^{x+y} \Rightarrow_{\mathsf{O}} \mathit{Suspended}^{(x+y,tran)})^{(2008,pers)})@(2008, pers)$$

The duration of tenure spanning from 2008 to 2012 is represented as follows:

$$r1: (\mathit{Elected}^{2008} \Rightarrow_{\mathsf{O}} \mathit{Tenure}^{(2008,pers)})^{(2008,pers)})@(2008, pers)$$
$$r2: (\mathit{Elected}^{2008} \rightsquigarrow_{\mathsf{O}} \neg \mathit{Tenure}^{2012})^{(2008,pers)})@(2008, pers)$$

Arts. 157-159 of the Italian Penal Code state the following:

$$\mathit{Art.\ 157}: (\mathit{Crime}^{x}, \mathit{Terms}^{x+y} \Rightarrow_{\mathsf{O}} \mathit{CrimeCancelled}^{(x+y,pers)})^{(z,pers)})@(z, pers)$$
$$\mathit{Art.\ 158}: (\mathit{Crime}^{x} \Rightarrow_{\mathsf{O}} \mathit{Terms}^{(x,pers)})^{(z,pers)})@(z, pers)$$
$$\mathit{Art.\ 159}: (\mathit{Crime}^{x}, \mathit{Suspended}^{x+y} \Rightarrow_{\mathsf{O}} \neg \mathit{Terms}^{(x+y,tran)})^{(z,pers)})@(z, pers)$$

As proposed by [16], the distinction between abrogation and annulment requires to distinguish between void *rules and* empty *rules. The content of a void rule, e.g.,* $(r\colon \bot)^t@t'$ *is* $\bot$*, while for the empty rule the value is the empty set. This means that the void rule has value for the combination of the temporal parameters, while for the empty rule, the content of the rule does not exist for the given temporal parameters.*

Given a rule $(r\colon A \Rightarrow b^{t_b})^{t_r}@t$*, the abrogation of* r *at* t_a *in repository* t' *is basically obtained by having in the theory the following meta-rule*

$$abr_r\colon \Rightarrow (r\colon \bot)^{(t_a,pers)})@(t', pers) \tag{10}$$

where $t' > t$*. The abrogation simply terminates the applicability of the rule. More precisely this operation sets the rule to the void rule. The rule is not removed from the system, but it has now a form where no longer can produce effects. In the case of the Legislative Act n. 124 (23 July 2008) we would have*

$$abr_{L.\ 124}\colon \Rightarrow (L.\ 124\colon \bot)^{(2011,pers)})@(2011, pers)$$

Hence, we can have the following, for example

- *at time* x*, from the viewpoint* x *we derive* $Suspended^x$*,* $2008 \leq x \leq 2010$*;*
- *at time* x*, from the viewpoint* x *we show that we cannot derive* $Terms^x$*,* $2008 \leq x \leq 2010$*;*
- *at time* 2011*, from viewpoint* 2011 *we show that we cannot derive* $Suspended^{2011}$*;*
- *at time* 2011*, from viewpoint* 2011 *we can derive* $Terms^{2011}$*.*

This is in contrast to what we do for annulment where the rule to be annulled is set to the empty rule. This essentially amounts to removing the rule from the repository. From the time of the annulment the rule has no longer any value. All past effects are thus blocked as well.

The definition of a modification function for annulment depends on the underlying variants of the logic, in particular whether conclusions persist across repositories. Minimally, the operation requires the introduction of a meta-rule setting the rule r *to be annulled to* $\emptyset$*, with the time when the rule is annulled and the time when the meta-rule is inserted in the legal system:*

$$(annul_r\colon \Rightarrow (r\colon \emptyset)^{(t_a,pers)})@(t', pers) \tag{11}$$

Hence,

$$(annul_{L.\ 124}\colon \Rightarrow (L.\ 124\colon \emptyset)^{(2008,pers)})@(2011, pers)$$

If we assume that conclusions persist over repositories we need some additional technical machinery to block pasts effects from previous repositories. In this case, since L. 124 *is modeled as a transient rule, we have basically to add a defeater like the following*[4]*:*

$$((annul_{ef}\colon \sim_{\mathsf{O}} \neg Suspended^{2008})^{(2008,pers)})@(2011, pers)$$

Hence, we now have, for example

- *we can show that we cannot derive at* x*, from viewpoint 2011,* $Suspended^x$*,* $2008 \leq x$*;*
- *we can prove at* x*, from viewpoint* 2011 $Terms^x$*,* $2008 \leq x$*.*

As stated before, another approach to address a logical model of norm change in the law is to enrich belief revision techniques by adding several temporal dimensions, as done in [29, 30]. There, techniques from belief revision formalisms are integrated with interval-based logical rules for legal systems, formalizing a revision operator. This operator may remove rules when needed or adapt intervals of time when contradictory norms are added in the system. This is discussed in the following section.

4 Temporalising Belief Revision for the Law

Example 1 involves information and rules referring to intervals of time in which some taxes applies. Cases like this, need to go beyond AGM machinery. Some research has been carried out to reframe AGM ideas, some of these, within richer rule-based logical systems [28, 26], and other, have aimed to study belief revision for situations in which nonmonotonic reasoning is addressed [33, 25]. However, also these attempts suffer from some drawbacks of standard AGM, among them the fact that the proposed frameworks fail to handle the temporal aspects of norm change.

Unlike rich but complex frameworks such as the one of [17]—which we have recalled in Section 3—we claim that belief revision techniques—which are based on an abstract and elegant machinery—can be reconciled with the need to consider several temporal patterns of legal reasoning. In [30] the authors are thus interested in the formalization of a belief revision operator applied to an epistemic model that considers rules and time. They enrich a simple logic language with an interval-based model of time, to represent temporal dimensions such as the effectiveness of norms,

[4]The general procedure to block conclusions when conclusions persist over repositories can be very complex: for all details, see [16].

i.e., when norms are applicable. There, the revision operator may remove rules when needed or adapt intervals of time when newer, contradictory norms are introduced in the system. In particular, the idea is the formalization of a belief revision operator that can address the evaluation of *timed rules* representing legal norms. Technical aspects of temporalised knowledge are considered in the following sections.

4.1 Legal System as Temporalised Belief Base

The problem of representing temporal knowledge and temporal reasoning arises in many disciplines, including Artificial Intelligence. A usual way to do this is to determine a *primitive* to represent time, and its corresponding *metric relations*. There are in the literature two traditional approaches to reasoning with and about time: a point based approach, as in [17], and an interval based approach as in [5, 12]. In the first case, the emphasis is put on *instants* of time (e.g., timestamps) and a relation of precedence among them. In the second case, time is represented as continuous sets of instants in which something relevant occurs. These intervals are identified by the starting and ending instants of time.

The approach introduced in [30], time intervals (like in [7, 12]) are considered. Following the semantics of the temporalised rules proposed in [17] and explained in Section 3 (in an adapted version), the revision operator, in essence, consists in the handling of intervals in order to maintain the consistency.

The above-mentioned temporal machinery is able to explicitly model two temporal dimensions among those mentioned above in Section 2.1, that is the time of norm effectiveness —i.e. when a norm can produce legal effects—and the time when the norm effects hold [17].

In [30], a propositional language $\mathbb{L}$ with a complete set of boolean connectives: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ is adopted and a consequence operator, denoted $Cn(\cdot)$, is used that takes sets of sentences in $\mathbb{L}$ and produces new sets of sentences. In general, in this article, we will write $\alpha \in Cn(A)$ as $A \vdash \alpha$.

Note that the AGM model [4] represents epistemic states by means of belief sets, that is, sets of sentences closed under logical consequence. Other models use belief bases; i.e., arbitrary sets of sentences [13, 20, 31]. In [30], epistemic model is based on an adapted version of belief bases which have additional information (time intervals).

4.2 Time Interval

In [30] a universal finite *set of time labels* $\mathbb{T} = \{t_1, \ldots, t_n\}$ strictly ordered is considered; each time label represents a unique time instant. Simplifying the notation,

$t_i - 1$ is the immediately previous instant to the instant t_i and $t_i + 1$ is the immediately posterior instant to the instant t_i. An interval is considered like a finite ordered sequence of time labels $t_i, \ldots, t_j$ where i, j are natural numbers ($i \leq j$) and $t_i, \ldots t_j \in \mathbb{T}$ denoting instances of time or *timepoints*. The discreteness of the flow of time is appropriate for modelling norms dynamics since norms usually refer to time in the spectrum of hours, days, months and years. Generally speaking, the law itself views time as determined by discrete steps. Thus, let $\alpha \in \mathbb{L}$, we have expressions of the type $\alpha^{interval}$, where *interval* can be as follow:

- $[t_i, t_i]$: meaning that α holds at time t_i. Following [17] α is transient (holding at precisely one instant of time). For simplicity $[t_i, t_i] = [t_i]$.
- $[t_i, \infty]$: meaning that α holds from t_i. Following [17] α is (indefinitely) persistent from t_i.
- $[t_i, t_j]$: meaning that α holds from time t_i to t_j with $t_i < t_j$.

Then a set of time intervals $\mathbb{I}$ contains intervals as those described previously. Thus, for simplicity, we can have expressions like α^J where $J \in \mathbb{I}$. Intervals in $\mathbb{I}$ will be denoted by uppercase Latin characters: $A, B, C, \ldots, Z$. Then, throughout this work α^J is a *temporalised sentence* meaning the sentence α has an effectiveness time indicated by J. Then the semantics of classical propositional logic to a timed context is preserved. A temporalised sentence $\alpha^{[t_a, t_b]}$ is true when its non-temporalised expression α is true in every time point t between t_a and t_b. In other words, α holds at $[t_a, t_b]$.

Naturally, two intervals may not be disjoint, as defined next.

Definition 1 (Contained interval)**.** *Let $R, S \in \mathbb{I}$ be two intervals. R is contained in S, denoted $R \subseteq S$ if and only if for all $t_i \in R$ it holds that $t_i \in S$.*

Definition 2 (Overlapped interval)**.** *Let $R, S \in \mathbb{I}$ be two intervals. R and S are overlapped, denoted $R \top S$ if and only if there exists $t_i \in R$ such that $t_i \in S$.*

Example 4. *Let $R, S, V \in \mathbb{I}$ where $R = [t_3, t_7]$, $S = [t_4, t_6]$ and $V = [t_5, t_9]$ with $t_3, t_4, t_5, t_6, t_7, t_9 \in \mathbb{T}$. Then $S \subseteq R$, $R \top V$ and $S \top V$.*

4.3 Temporalised Belief Base

As rules are part of the knowledge, they are subject of temporal effectiveness too. In this perspective, there may be expressions such as

$$\alpha^{[t_a, t_b]} \rightarrow \beta^{[t_c, t_d]}$$

meaning that the rule can derive that β holds from time t_c to t_d if it is proved that α holds from time t_a to t_b. The notion of persistence during a given interval could be also applied to rules, although we adopt here a general approach. Note that the above implication itself is not decorated with intervals, but α and β are. This means that the implication always holds at $[-\infty, \infty]$ and hence again the classical semantics of first order logic is preserved. Thus, if the implication holds (since it is not conditioned in time) and α holds at $[t_a, t_b]$ then β holds at $[t_c, t_d]$.

Example 5. *The provision from Example 1 "If unemployed from 1980 to 1983, then a tax exemption applies from 1984 to 1986" can be formalised as follows:*

$$Unemployed^{[1980,1983]} \rightarrow Tax\_Exemption^{[1984,1986]}.$$

Thus, in [29], *temporalised belief base* which will contain temporalised sentences (see Example 6) is defined. This base represents a legal system in which each temporalised sentence defines a norm whose time interval determines the effectiveness time.

Example 6. *The set*

$$\begin{aligned}\mathbb{K} = \{&\alpha^{[t_1,t_3]}, \alpha^{[t_4]}, \alpha^{[t_1,t_4]} \rightarrow \beta^{[t_4,t_6]},\\ &\beta^{[t_5,t_6]}, \beta^{[t_6,t_8]}, \beta^{[t_{10}]}, \delta^{[t_{11}]},\\ &\delta^{[t_{11}]} \rightarrow \beta^{[t_{15},t_{20}]}, \omega^{[t_2,t_8]},\\ &\omega^{[t_4]} \rightarrow \beta^{[t_6,\infty]}, \epsilon^{[t_1,\infty]}\}\end{aligned}$$

is a valid temporalised belief base for a legal system. Note that sentence ϵ is valid (or true) from t_1.

This type of belief base representation implies that a sentence can appear more than once in a temporalised belief base, but from the point of view of the temporalised sentences stored in the temporalised belief base there is no redundancy because each temporalised sentence has different time intervals. For instance, consider Example 6, where α and β appear twice, but with different intervals. Whenever a sentence appears more than once with different intervals, just like $\beta^{[t_5,t_6]}$ and $\beta^{[t_{10}]}$, this sentence is said to be **intermittent**. Also note that if the intervals of a sentence are overlapped or continuous through the knowledge base, like $\beta^{[t_5,t_6]}$, $\beta^{[t_6,t_8]}$ in Example 6, the different occurrences are not collapsed into one, producing $\beta^{[t_5,t_8]}$. This is for two main reasons. First, a knowledge base scanning procedure is needed for identifying overlapped or continuous temporalised sentences, adding extra, yet small, complexity which is not relevant for the belief revision operator discussed

here. Second, in law this kind of reiteration of a sentence in different intervals is not uncommon. For instance, the government may decide that there is a tax exemption during quarantine in March, and some weeks later then decide that the same exemption also holds during April. Here there are two legal norms that conform a continuous benefit, but with separate identities that can be revised for different reasons. Even more, the continuity does not need to be so explicit: two different continuous sentences like $\alpha^{[t_1,t_n]}$, $\alpha^{[t_{n+1},t_m]}$ may be derived from different, separate portions of the knowledge base, and even when it is clear that α holds from t_1 to t_m, this wider interval will not be derived as it is. That is, as we will see below, a sentence can be implicitly represented on a belief base by several different derivations that maintain the validity of the sentence at overlapping intervals. In this case, it could not be explicitly represented with a single sentence the validity of it at all times.

4.4 Temporalised Derivation

Note that a norm can explicitly be in a temporalised belief base, as $\alpha^{[t_5]} \in \mathbb{K}$ in Example 6. However, a norm can implicitly be represented in a temporal belief base if some conditions hold. For instance, in Example 6, norm β is implicitly represented with $\omega^{[t_2,t_8]}$, $\omega^{[t_4]} \rightarrow \beta^{[t_6,\infty]}$ due to the antecedent of the rule is held in t_4 by the temporalised sentence $\omega^{[t_2,t_8]}$. Next, the notion of temporalised derivation for a sentence is introduced to capture this intuition. To do this, we first give a definition of temporalised derivation in a time instant and then we give a definition of temporalised derivation in time interval.

Definition 3 (Temporalised derivation in a time instant)**.** *Let $\mathbb{K}$ be a set of temporalised sentences and $\alpha^{[t_i]}$ be a temporalised sentence. We say that $\alpha^{[t_i]}$ is derived from $\mathbb{K}$, denoted $\mathbb{K} \vdash^t \alpha^{[t_i]}$, if and only if:*

- *$\alpha^J \in \mathbb{K}$ and $t_i \in J$, or*
- *$\beta^H \rightarrow \alpha^P \in \mathbb{K}$ and $t_i \in P$ and $\mathbb{K} \vdash^t \beta^{[t_j]}$ for all $t_j \in H$.*

Definition 4 (Temporalised derivation in a time interval)**.** *Let $\mathbb{K}$ be a set of temporalised sentences and $\alpha^{[t_i,t_j]}$ be a temporalised sentence. We say that $\alpha^{[t_i,t_j]}$ is derived from $\mathbb{K}$ (denoted $\mathbb{K} \vdash^t \alpha^{[t_i,t_j]}$) if and only if $\mathbb{K} \vdash^t \alpha^{[t_p]}$ for all $t_p \in [t_i, t_j]$.*

Computing the temporalised derivation of a sentence through checking each instant of the intervals is useful in special cases where implicit sentences need temporalised sentences with overlapped intervals as antecedents. To determine the time interval of the implicitly derived temporal sentence, the temporal consequence will be defined below.

Definition 5 (Temporalised consequence)**.** *Let* $\mathbb{K}$ *be a set of temporalised sentences and* $\alpha^{[t_i,t_j]}$ *be a temporalised sentence. We say that* $\alpha^{[t_i,t_j]}$ *is a temporalised consequence of* $\mathbb{K}$ *(*$\alpha^{[t_i,t_j]} \in Cn^t(\mathbb{K})$*) if and only if* $\mathbb{K} \vdash^t \alpha^{[t_i,t_j]}$*.*

Example 7. *Consider again the temporalised belief base of Example 6. Then,* $\mathbb{K} \vdash^t \beta^{[t_4,\infty]}$*, that is,* $\beta^{[t_4,\infty]} \in Cn^t(\mathbb{K})$*; and* $\mathbb{K} \vdash^t \alpha^{[t_1,t_4]}$*, that is,* $\alpha^{[t_1,t_4]} \in Cn^t(\mathbb{K})$*.*

Following Definition 4, notice that the **interval of an implicitly derived sentence** will be the interval of the consequent of the rule that derives the conclusion of the proof. For instance, suppose that $\mathbb{K} = \{\gamma^{[t_2,t_5]}, \gamma^{[t_3,t_4]} \rightarrow \epsilon^{[t_6,t_9]}\}$ then the time interval of ϵ is $[t_6, t_9]$. Thus, a temporalised sentence $\alpha^{[t_i,t_j]}$ is **valid** (or true) in $\mathbb{K}$ if $\mathbb{K} \vdash^t \alpha^{[t_i,t_j]}$.

Thus, in [30], a ***contradiction*** arises when two complementary sentences can be derived with time intervals overlapped. For instance, suppose $\mathbb{K} = \{\alpha^{[t_2,t_9]}, \neg\alpha^{[t_1,t_3]}\}$, in this case, there exists a contradiction. However, consider $\mathbb{K} = \{\alpha^{[t_5]}, \neg\alpha^{[t_1,t_3]}\}$, in this case, we will say that $\mathbb{K}$ does not have contradictions. Moreover, a temporalised belief base is ***temporally consistent*** if the base does not have contradictions. The temporalised belief base of Example 6 is temporally consistent.

Remark 1. *If* $\mathbb{K}$ *represents a legal system then* $\mathbb{K}$ *should be temporally consistent.*

4.5 Legal Belief Revision

From a rational point of view, as was mentioned in Remark 1, a legal system should be temporally consistent, i.e., it cannot contain contradictory norms at any time. Hence, in [30], the authors propose a **prioritised legal revision operator** that allows to consistently add a temporalised sentence $\alpha^{[t_i,t_j]}$ to a consistent legal system $\mathbb{K}$.

This special revision operator is inspired by the rule semantics explained above in Section 4.1 (an adapted version from the one proposed in [17]). Thus, following the concept of temporally consistency of Subsection 4.4, the revision operator may remove temporalised sentences or, in some cases, may only modify the intervals to maintain consistency.

To incorporate a norm $\neg\beta^J$ into a legal system, it is necessary to consider all possible contradictions that may arise if the norm is added without checking for consistency. For this reason, it is necessary to compute all proofs of β considering only those temporalised sentences β^P whose effectiveness time is overlapped with the time interval J, that is, $J \top P$. Note that it is optimal to compute all minimal proofs of a temporal sentence considering only those in which the time interval is overlapped with the time interval of the input sentence. Next, a set of minimal proofs for a sentence is defined.

Definition 6 (Minimal proof)**.** *Let $\mathbb{K}$ be a temporalised belief base and α^J a temporalised sentence. Then, $\mathbb{H}$ is a minimal proof of α^J if and only if*

1. $\mathbb{H} \subseteq \mathbb{K}$,
2. $\alpha^P \in Cn^t(\mathbb{H})$ *with* $J \top P$, *and*
3. *if* $\mathbb{H}' \subset \mathbb{H}$, *then* $\alpha^P \notin Cn^t(\mathbb{H}')$ *with* $J \top P$.

Given a temporalised sentence α^J, the function $\Pi(\alpha^J, \mathbb{K})$ returns the set of all the minimal proofs for α^J from $\mathbb{K}$.

Remark 2. *Each set of $\Pi(\alpha^J, \mathbb{K})$ derives α in at least one time instant of J.*

Example 8. *Consider the temporalised belief base of Example 6. Then $\Pi(\beta^{[t_5,t_6]}, \mathbb{K})$ $= \{\mathbb{H}_1, \mathbb{H}_2, \mathbb{H}_3, \mathbb{H}_4\}$ where:*

- $\mathbb{H}_1 = \{\alpha^{[t_1,t_3]}, \alpha^{[t_4]}, \alpha^{[t_1,t_4]} \rightarrow \beta^{[t_4,t_6]}\}$,
- $\mathbb{H}_2 = \{\beta^{[t_5,t_6]}\}$,
- $\mathbb{H}_3 = \{\beta^{[t_6,t_8]}\}$,
- $\mathbb{H}_4 = \{\omega^{[t_2,t_8]}, \omega^{[t_4]} \rightarrow \beta^{[t_6,\infty]}\}$

Note that $\mathbb{H}_1$ is minimal: α should be derived from t_1 to t_4 to use the rule $\alpha^{[t_1,t_4]} \rightarrow \beta^{[t_4,t_6]}$ hence, $\alpha^{[t_1,t_3]}$ and $\alpha^{[t_4]}$ should be in $\mathbb{H}_1$.

The construction of prioritised legal revision by a temporalised sentence is based on the concept of a minimal proof; to complete the construction, an incision function is used which selects in every minimal proof the sentence to be erased later and which can produce legal effects in favour of a possible contradiction with the new norm.

The operator is based on a selection of sentences in the knowledge base that are relevant to derive the sentence to be retracted or modified. In order to perform a revision, following kernel contractions [21], this approach uses *incision functions*, which select from the minimal subsets entailing the piece of information to be revoked or modified. An incision function only selects sentences that can be relevant for α and at least one element from each $\Pi(\alpha^J, \mathbb{K})$:

Definition 7 (Incision function)**.** *Let $\mathbb{K}$ be a temporalised belief base. An incision function σ for $\mathbb{K}$ is a function such that for all $\alpha^J \in Cn^t(\mathbb{K})$:*

- $\sigma(\Pi(\alpha^J, \mathbb{K})) \subseteq \bigcup(\Pi(\alpha^J, \mathbb{K}))$.
- *For each* $\mathbb{H} \in \Pi(\alpha^J, \mathbb{K})$, $\mathbb{H} \cap \sigma(\Pi(\alpha^J, \mathbb{K})) \neq \emptyset$.

In Hansson's approach it is not specified how the incision function selects the sentences that will be discarded of each minimal proof. In this approach, this is solved by considering those sentences that can produce legal effects in favour of a possible contradiction with the new norm. Thus, if the new norm is $\neg\beta^J$ then the incision function selects the temporalised sentences β^P or $\alpha^Q \rightarrow \beta^F$ of each $\mathit{\Pi}(\beta^J, \mathbb{K})$.

Definition 8 (Search consequence function)**.** *Sc:* $\mathbb{L} \times \mathbb{K} \mapsto \mathbb{K}$*, is a function such that for a given sentence* α *and a given temporalised base* $\mathbb{K}$ *with* $\mathbb{H} \subseteq \mathbb{K}$*,*

$$Sc(\alpha, \mathbb{H}) = \{\alpha^J : \alpha^J \in \mathbb{H}\} \cup \{\beta^P \rightarrow \alpha^Q : \beta^P \rightarrow \alpha^Q \in \mathbb{H} \text{ and } \beta \in \mathbb{L}\}.$$

Definition 9 (Consequence incision function)**.** *Given a set of minimal proofs* $\mathit{\Pi}(\alpha^J, \mathbb{K})$*, $\sigma^c$is a consequence incision function if it is a incision function for* $\mathbb{K}$ *such that*

$$\sigma^c(\mathit{\Pi}(\alpha^J, \mathbb{K})) = \bigcup_{\mathbb{H} \in \mathit{\Pi}(\alpha^J, \mathbb{K})} Sc(\alpha, \mathbb{H}).$$

Example 9. *Consider Examples 6 and 8. Then,* $Sc(\beta, \mathbb{H}_1) = \{\alpha^{[t_1,t_4]} \rightarrow \beta^{[t_4,t_6]}\}$*,* $Sc(\beta, \mathbb{H}_2) = \{\beta^{[t_5,t_6]}\}$*,* $Sc(\beta, \mathbb{H}_3) = \{\beta^{[t_6,t_8]}\}$*, and* $Sc(\beta, \mathbb{H}_4) = \{\omega^{[t_4]} \rightarrow \beta^{[t_6,\infty]}\}$*. Then*

$$\begin{aligned}\sigma^c(\mathit{\Pi}(\beta^{[t_5,t_6]}, \mathbb{K})) &= \bigcup_{\mathbb{H} \in \mathit{\Pi}(\beta^{[t_5,t_6]}, \mathbb{K})} Sc(\beta, \mathbb{H}) \\ &= \{\alpha^{[t_1,t_4]} \rightarrow \beta^{[t_4,t_6]}, \beta^{[t_5,t_6]}, \beta^{[t_6,t_8]}, \omega^{[t_4]} \rightarrow \beta^{[t_6,\infty]}\}\end{aligned}$$

As mentioned before, the revision operator may remove temporalised sentences or, in some cases, may modify the intervals to maintain consistency. Next, a temporal projection will be introduced based on a given time interval. The idea here is, given a temporalised belief base $\mathbb{K}$ and given a time interval $[t_i, t_j]$, to return a temporalised belief base $\mathbb{K}'$ containing those sentences from $\mathbb{K}$ whose time intervals be out of $[t_i, t_j]$.

Definition 10 (Excluding temporal projection)**.** *Let* $\mathbb{K}$ *be a temporalised belief base and let* $[t_i, t_j]$ *be a time interval where* $t_i, t_j \in \mathbb{T}$*. A excluding temporal projection of* $\mathbb{K}$ *from* t_i *to* t_j*, denoted out*$(\mathbb{K}, [t_i, t_j])$*, is a subset of* $\mathbb{K}$ *where for all* $\alpha^{[t_p,t_q]} \in \mathbb{K}$*, out*$(\mathbb{K}, [t_i, t_j])$ *will contain:*

- $\alpha^{[t_p,t_i-1]}$ *if* $t_p < t_i$*,* $t_q \geq t_i$ *and* $t_q \leq t_j$*,*
- $\alpha^{[t_j+1,t_q]}$ *if* $t_p \geq t_i$*,* $t_q > t_j$ *and* $t_p \leq t_j$*,*

- $\alpha^{[t_p,t_i-1]}$ *and* $\alpha^{[t_j+1,t_q]}$ *if* $t_p < t_i$, $t_q > t_j$,
- $\alpha^{[t_p,t_q]}$ *if* $t_q < t_i$ *or* $t_p > t_j$.

Remark 3. *Note that when* $t_p \geq t_i$ *and* $t_q \leq t_j$, *the temporal sentence is not considered. In this case, this sentence is erased.*

Remark 4. *Note that if* $\delta^{[t_h,t_k]} \in out(\mathbb{K}, [t_i, t_j])$ *and the interval* $[t_h, t_k]$ *is generated through excluding temporal projection of* $\mathbb{K}$ *from* t_i *to* t_j *then there exists a temporal sentence* $\delta^{[t_p,t_q]}$ *in* $\mathbb{K}$ *such that* $[t_h, t_k] \subseteq [t_p, t_q]$.

Example 10. *Consider Example 9 and suppose that* S *is a temporalised belief base and* $S = \sigma^c(\Pi(\beta^{[t_5,t_6]}, \mathbb{K}))$. *Then,* $out(S, [t_5, t_6]) = \{\alpha^{[t_1,t_4]} \rightarrow \beta^{[t_4]}, \beta^{[t_7,t_8]}, \omega^{[t_4]} \rightarrow \beta^{[t_7,\infty]}\}$.

Following the notion of excluding temporal projection (Definition 10) a norm prioritised revision operator is defined. That is, an operator that allows to *consistently* add temporalised sentences in a temporalised belief base. If a contradiction arises, then the revision operator may remove temporalised sentences or modify the corresponding intervals in order to maintain consistency.

Definition 11. *Let* $\mathbb{K}$ *be a temporalised belief base and* α^J *be a temporalised sentence. The operator "* $\otimes$*", called prioritised legal revision operator, is defined as follow:*

$$\mathbb{K} \otimes \alpha^J = (\mathbb{K} \setminus \sigma^c(\Pi(\neg\alpha^J, \mathbb{K}))) \cup out(\sigma^c(\Pi(\neg\alpha^J, \mathbb{K})), J) \cup \{\alpha^J\}.$$

Note that, to add α^J to $\mathbb{K}$, all temporized sentences that have $\neg\alpha$ as a consequence and contribute to derive some instant of $\neg\alpha^J$ are erased. Then, these same sentences are added but with their modified intervals (using the excluding temporal projection introduced in Definition 10). Finally, α^J is added.

Example 11. *Consider Example 6 and suppose that a new norm* $\neg\beta^{[t_5,t_6]}$ *it is wished to add. To do this, it is necessary to do* $\mathbb{K} \otimes \neg\beta^{[t_5,t_6]}$. *Consider Examples 8 and 9. Then,* $\mathbb{K} \otimes \neg\beta^{[t_5,t_6]} = \{\alpha^{[t_1,t_3]}, \alpha^{[t_4]}, \alpha^{[t_1,t_4]} \rightarrow \beta^{[t_4]}, \beta^{[t_7,t_8]}, \beta^{[t_{10}]}, \delta^{[t_{11}]}, \delta^{[t_{11}]} \rightarrow \beta^{[t_{15},t_{20}]}, \omega^{[t_2,t_8]}, \omega^{[t_4]} \rightarrow \beta^{[t_7,\infty]}, \epsilon^{[t_1,\infty]}, \neg\beta^{[t_5,t_6]}\}$. *Note that, this new temporalised base is temporally consistent.*

The following example shows how our operator works in a particular situation when a legal system undergoes many changes and has rules that complement each other.

Example 12. *Consider the following temporalised belief base* $\mathbb{K} = \{\beta^{[t_1,t_{10}]}, \beta^{[t_1,t_5]} \rightarrow \alpha^{[t_1,t_5]}, \beta^{[t_6,t_{10}]} \rightarrow \alpha^{[t_6,t_{10}]}, \delta^{[t_4]}\}$. *Note that,* $\mathbb{K} \vdash^t \alpha^{[t_1,t_{10}]}$ *because* $\mathbb{K} \vdash^t \alpha^{[t_i]}$ *for all* $t_i \in [t_1, t_{10}]$. *Suppose that it is necessary to adopt* $\neg\alpha^{[t_1,t_{10}]}$. *To do this, it is necessary to compute all the minimal proofs of* $\alpha^{[t_1,t_{10}]}$ *in* $\mathbb{K}$. *In this case,* $\Pi(\alpha^{[t_1,t_{10}]}, \mathbb{K}) = \{\{\beta^{[t_1,t_{10}]}, \beta^{[t_1,t_5]} \rightarrow \alpha^{[t_1,t_5]}, \beta^{[t_6,t_{10}]} \rightarrow \alpha^{[t_6,t_{10}]}\}\}$. *Then,* $S = \sigma^c(\Pi(\alpha^{[t_1,t_{10}]}, \mathbb{K})) = \{\beta^{[t_1,t_5]} \rightarrow \alpha^{[t_1,t_5]}, \beta^{[t_6,t_{10}]} \rightarrow \alpha^{[t_6,t_{10}]}\}$. *Thus,* $out(S, [t_1, t_{10}]) = \emptyset$. *Therefore,* $\mathbb{K} \otimes \neg\alpha^{[t_1,t_10]} = \{\beta^{[t_1,t_{10}]}, \delta^{[t_4]}, \neg\alpha^{[t_1,t_{10}]}\}$.

4.6 Others works that have discussed the relation between belief revision and temporal reasoning

There are some works in the literature that have discussed the relation between belief revision and temporal reasoning, though none of them addressed the issue in the normative domain. Two prominent lines of investigation are [10, 11] and [27].

[10, 11] address belief revision in a temporal logic setting. These articles consider sets of sentences closed under logical consequence. In contrast to this, the approach proposed in [30] is based on an adapted version of belief bases which have additional information (time intervals). The use of belief bases makes the representation of the legal system state more natural and computationally tractable. That is, following [22, page 24] and [31], it is considered that legal systemsâĂŹ sentences could be represented by a finite number of sentences that correspond to the explicit beliefs on the legal system. The main purpose of [10, 11] is to represent the AGM postulates as axioms in a modal language. The assumption is that belief revision has to do with the interaction of belief and information over time, thus temporal logic seemed a natural starting point. The technical solution is to consider branching-time frames to represent different possible evolutions of beliefs. Hence, belief revision operators are interpreted over possible worlds. Unlike this, the authors in [30] work with legal system in which each temporalised sentence defines a norm whose time interval determines the effectiveness time. Then, the revision process defined in [30] may remove temporalised sentences or, in some cases, may only modify the intervals.

[27] is based on a well-developed theory of action in the situation calculus extended to deal with belief. The authors add to this framework a notion of plausibility over situations, and show how to handle nested belief, belief introspection, mistaken belief, belief revision and belief update together with iterated belief change.

An interesting line of investigation is to study possible correlations with these two last research lines in literature as compared to the system proposed in [30]. Such a comparison cannot be directly done from technical viewpoint for two reasons. First of all, [30] is specifically focused in a propositional language following kernel contraction construction proposed in [21]. Second, the propositional language in

[30] is equipped with explicit time-stamps and with temporal intervals, which allow them for expressing richer temporal specifications in the language.

5 Conclusions

In order to properly model norm change in the law, temporal aspects of legal dynamics must be considered. Several reasons support this idea. The law regulates its own changes by stating, within the system, what and how other existing norms should be modified. The introduced new norms can be in conflict and so norms are defeasible by nature. Even more, legal norms are qualified by diverse temporal properties, such as the time when the norm is added to the legal system, or when the norm is in force and it produces legal effects. Thus, all these aspects may be addressed by two different pathways, as reflected in the literature. First, normative dynamics can be modelled by combining logical systems for temporal and defeasible reasoning [15, 19, 16]. Second, belief revision techniques can be enriched with temporal dimensions: this has been done in works such as [29, 30]. These are two different approaches to the consideration of time within a logical framework for legal dynamics.

Defeasible Logic was extended with temporal parameters to allow for reasoning about time specified inside norms. Two temporal dimensions are considered: the first one is when the norm is in force in a normative system, and the second is when the norm exists in the normative system from a certain viewpoint. Usually only the time of force is considered, but here the notion of *temporalised rule with viewpoint* is introduced, a mechanism through which it is possible to reason on the legal system from the viewpoint of its current version but as if it were revised in the past. This extension increases the expressive power of the logic and it allows us to represent meta-norms describing norm modifications by referring to a variety of possible timelines through which conclusions, rules and derivations can persist over time. This formalism has been shown useful to model retroactive legal modifications, a complex timed behaviour of legal systems that requires special attention. Hence, this model is suitable for modeling changes that affect the legal system with respect to legal effects which were also obtained before the legal change was done. This is not a simple feature and the formalism addresses it properly.

On the other hand, a contrasting approach explores the importance of time in legal dynamics from the point of view of *revision of beliefs* in laws. This make sense since the law is a *dynamic* system of rules. Indeed, a very complex one: as times goes by, rules are introduced in the system, which may be either unexpectedly in conflict with existing rules or be intended to provide new, different norms for society.

This demands a consistent revision of the rules of the system, so an extension of classic belief revision formalism seems to be appropriate. Then, we discussed here the second approach, which proposes a belief revision operator that considers time interval in the revision process. Intervals are used to model a period of time for a piece of knowledge to be effective or relevant, leading to the definition of a new kind of temporal rules. On these interval-decorated rules the corresponding temporalised derivation was defined. The consideration of time requires an adaptation of the notions of contradiction and inconsistency in the classical sense. Temporalised knowledge base is inconsistent only if contradictory information can be derived for the same moment of time. In that approach was defined a novel belief revision operator that allows the consistent addition of temporalised sentences in a temporalised belief base. If a contradiction arises, then the revision operator may either completely remove conflictive temporalised sentences or modify the intervals of some rules. This last action is made because a given consequence a at interval I may fall in contradiction during a sub-interval of I. Thus, a should be a consequence, after the revision, only for the rest of I. Then, intervals in rules should be taken into account for the revision process.

The central idea of this research topic is that formal models of norm change must address the fact that new norms may be elicited and old norms may need to be retracted, with complex consequences. Depending on the particular feature of legal dynamics intended to be modelled, any proposed framework requires an appropriate model of time. There are two mainstream approaches to reasoning with and about time: point based and interval based. Here we explored both flavours, by discussing two different, interesting approaches to the consideration of time within the study of legal dynamics. Both formalisms take time into account and provide the necessary logical infrastructure to address the characterization of complex behaviour of dynamics in normative systems, constituting solid foundations for further research.

References

[1] Carlos E. Alchourrón and Eugenio Bulygin. The expressive conception of norms. In Risto Hilpinen, editor, *New Studies in Deontic Logic*, pages 95–125. D. Reidel, Dordrecht, 1981.

[2] Carlos E. Alchourrón and David C. Makinson. Hierarchies of regulations and their logic. In Risto Hilpinen, editor, *New Studies in Deontic Logic*, pages 125–148. D. Reidel, Dordrecht, 1981.

[3] Carlos E. Alchourrón and David C. Makinson. The logic of theory change: Contraction functions and their associated revision functions. *Theoria*, 48:14–37, 1982.

[4] C.E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.

[5] James F. Allen. Towards a general theory of action and time. *Artif. Intell.*, 23(2):123–154, 1984.

[6] Grigoris Antoniou, David Billington, Guido Governatori, and Michael J. Maher. Representation results for defeasible logic. *ACM Trans. Comput. Log.*, 2(2):255–287, 2001.

[7] Juan Carlos Augusto and Guillermo Ricardo Simari. Temporal defeasible reasoning. *Knowl. Inf. Syst.*, 3(3):287–318, 2001.

[8] G. Boella, G. Pigozzi, and L. van der Torre. A normative framework for norm change. In *Proc. AAMAS 2009*, pages 169–176. ACM, 2009.

[9] Guido Boella, Gabriella Pigozzi, and Leon van der Torre. AGM contraction and revision of rules. *Journal of Logic, Language and Information*, 25(3-4):273–297, 2016.

[10] Giacomo Bonanno. Axiomatic characterization of the AGM theory of belief revision in a temporal logic. *Artif. Intell.*, 171(2-3):144–160, 2007.

[11] Giacomo Bonanno. Belief revision in a temporal framework. In *New Perspectives on Games and Interaction, volume 4 of Texts in Logic and Games*, pages 45–80. Amsterdam University Press, 2009.

[12] Maximiliano Celmo Budán, Maria Laura Cobo, Diego C. Martínez, and Guillermo Ricardo Simari. Bipolarity in temporal argumentation frameworks. *Int. J. Approx. Reasoning*, 84:1–22, 2017.

[13] André Fuhrmann. Theory contraction through base contraction. *Journal of Philosophical Logic*, 20(2):175–203, may 1991.

[14] Dov M. Gabbay, Gabriella Pigozzi, and John Woods. Controlled revision - an algorithmic approach for belief revision. *J. Log. Comput.*, 13(1):3–22, 2003.

[15] G. Governatori, M. Palmirani, R. Riveret, A. Rotolo, and G. Sartor. Norm modifications in defeasible logic. In *JURIX 2005*, pages 13–22. IOS Press, Amsterdam, 2005.

[16] G. Governatori and A. Rotolo. Changing legal systems: Legal abrogations and annulments in defeasible logic. *Logic Journal of IGPL*, 18(1):157–194, 2010.

[17] Guido Governatori and Antonino Rotolo. Changing legal systems: legal abrogations and annulments in defeasible logic. *Logic Journal of the IGPL*, 18(1):157–194, 2010.

[18] Guido Governatori, Antonino Rotolo, Francesco Olivieri, and Simone Scannapieco. Legal contractions: a logical analysis. In *Proc. ICAIL 2013*, 2013.

[19] Guido Governatori, Antonino Rotolo, Régis Riveret, Monica Palmirani, and Giovanni Sartor. Variants of temporal defeasible logics for modelling norm modifications. In *The Eleventh International Conference on Artificial Intelligence and Law, Proceedings of the Conference, June 4-8, 2007, Stanford Law School, Stanford, California, USA*, pages 155–159, 2007.

[20] Sven Ove Hansson. In defense of base contraction. *Syntheses*, 91(3):239–245, june 1992.

[21] Sven Ove Hansson. Kernel Contraction. *The Journal of Symbolic Logic*, 59:845–859, 1994.

[22] Sven Ove Hansson. *A Textbook of Belief Dynamics: Theory Change and Database Updating*. Kluwer Academic Publishers, 1999.

[23] H. L. A. Hart. *The Concept of Law*. Clarendon Press, Oxford, 1994.

[24] Hans Kelsen. *General theory of norms*. Clarendon, Oxford, 1991.

[25] Patrick Krümpelmann and Gabriele Kern-Isberner. Belief base change operations for answer set programming. In *Logics in Artificial Intelligence - 13th European Conference, JELIA 2012, Toulouse, France, September 26-28, 2012. Proceedings*, pages 294–306, 2012.

[26] Antonino Rotolo. Retroactive legal changes and revision theory in defeasible logic. In G. Governatori and G. Sartor, editors, *DEON 2010*, volume 6181 of *LNAI*, pages 116–131. Springer, 2010.

[27] Steven Shapiro, Maurice Pagnucco, Yves LespÃľrance, and Hector J. Levesque. Iterated belief change in the situation calculus. *Artificial Intelligence*, 175(1):165 – 192, 2011. John McCarthy's Legacy.

[28] Audun Stolpe. Norm-system revision: theory and application. *Artif. Intell. Law*, 18(3):247–283, 2010.

[29] Luciano H. Tamargo, Diego C. Martinez, Antonino Rotolo, and Guido Governatori. Temporalised belief revision in the law. In Adam Z. Wyner and Giovanni Casini, editors, *Legal Knowledge and Information Systems - JURIX 2017: The Thirtieth Annual Conference, Luxembourg, 13-15 December 2017.*, volume 302 of *Frontiers in Artificial Intelligence and Applications*, pages 49–58. IOS Press, 2017.

[30] Luciano H. Tamargo, Diego C. Martinez, Antonino Rotolo, and Guido Governatori. An axiomatic characterization ofÂătemporalised belief revision inÂătheÂălaw. *Artificial Intelligence and Law*, pages 1–21, 2019.

[31] Renata Wassermman. Resource bounded belief revision. *PhD Thesis, Institute for Logic, Language and Computation (ILLC), University of Amsterdam*, 2000.

[32] Gregory R. Wheeler and Marco Alberti. No revision and no contraction. *Minds and Machines*, 21(3):411–430, 2011.

[33] Zhiqiang Zhuang, James P. Delgrande, Abhaya C. Nayak, and Abdul Sattar. Reconsidering agm-style belief revision in the context of logic programs. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, volume 285 of *Frontiers in Artificial Intelligence and Applications*, pages 671–679. IOS Press, 2016.

 Received 1 February 2021

Computational Complexity of Compliance and Conformance: Drawing a Line Between Theory and Practice

Silvano Colombo Tosatto
Data61, CSIRO
silvano.colombotosatto@data61.csiro.au

Guido Governatori
Data61, CSIRO
guido.governatori@data61.csiro.au

Abstract

In the present chapter we focus our attention on the computational complexity of proving regulatory compliance of business process models. While the topic has never received the deserved attention, we argue that the theoretical results, both existing and yet to find, are far reaching for many areas related to the problem of proving compliance of process models. Therefore, we provide here and discuss the existing results concerning the theoretical computational complexity of the problem, as well as discussing some further areas that can potentially advance the knowledge about the issue, and other closely related disciplines that can either bring or take insights to this area.

1 Introduction

In this chapter we investigate the computational complexity of the problem of proving regulatory compliance of process models. This problem consists of verifying whether a process model, representing a set of executions of an organisation's procedures, complies with some given regulations. We consider an execution to be compliant with some regulations when no violations occurs in such execution with respect of the regulations. Different degrees of compliance are determined depending on whether every execution in a model comply with the regulations, when some comply, and when none comply.

Proving regulatory compliance has been receiving more attention during the past years, as shown by approaches of varying complexity being proposed in the literature (Some of the recently proposed approaches: [41, 34, 35]). Analysis of the current expenses [57] from real businesses and companies towards showing their compliance with the relevant regulations, have brought an interests in finding automated solutions in order to bring down such costs. For a recent survey of the approaches to business process compliance and open research question in the field see Hashmi et al. [37].

Despite the various approaches proposed to address the problem of proving the regulatory compliance of business process models, meaning "ensuring that, executing such a business process model to achieve a business objective, is compliant with the regulations", or dealing with conformance: "verifying whether existing executions, usually logs, have been performed in accordance to the regulatory requirements", in general, the computational complexity of the problem it has been for the most part ignored. Despite knowing that in general the problem is NP-complete [10], many approaches have shown to be able to solve current real problem without being hindered by the theoretical complexity of it. While this allows currently to use this kind of solutions without any sort of big issue, due to the current race towards automation, it is only bound that the business process models and the regulatory frameworks required to be verified in the future are increasing in size and complexity. This, in turn could potentially put a hard stop to the approaches currently used, due to them ignoring such theoretical complexity concerns.

In this chapter we first provide a computational complexity analysis of the general problem of proving regulatory compliance of business process models, and its variants obtainable by manipulating the properties of the regulatory framework being used. We consider three different properties that a regulatory framework can have: the number of regulations contained in the framework, whether the regulations affect the entirety of the executions of a business process model, or whether some parts of them given some additional conditions. The third property concerns whether the regulations are expressed using atomic boolean propositions, or full formulae. Given these binary properties we identify 8 variants of the problems, for which we study and provide their computational complexity classes.

Additionally, the computational complexity of the problem can change depending on the features of the business process models being verified. Taking as the basic variant in this scenario structured business process models, namely process models whose structure can be defined as a properly nested structure, which has technical advantages over processes not following such constraints, which in turn ends up being more expressive. We consider some additional features that can be desirable to represent real life processes, such as unstructured process models and the inclusion

of loops, and we discuss how these additions influence the computational complexity of the problem's variants.

After having discussed the theoretical computational complexity of the problem, we consider some of the existing approaches aimed at solving the problem of proving regulatory compliance of business process models, and we assign them to the problem variants identified in this chapter, hence associating them to a computational complexity class. Starting form this classification, we provide a preliminary study of the behaviour of these approaches in a future where the components of the problem increase in size and complexity, namely the business process models and / or the regulatory frameworks. We aim with this preliminary analysis to provide some insights concerning which approaches may be hindered by the theoretical complexity limitations of the problem as bigger and more complex problems will be required to be solved, and which may be potentially be still used to tackle these larger problems.

Furthermore, we discuss a problem related to the problem of proving regulatory compliance: conformance, discussing a few of the techniques used to solve this tangential problem.

Finally, we conclude this chapter by discussing some of the open problem concerning the computational complexity analysis of the problem of proving regulatory compliance of business process models.

2 The Problem: Proving Regulatory Compliance

In this section, we introduce the problem of proving regulatory compliance of business process models analysed in this chapter. The problem consists of two components:

i) the business process model compactly describing a set of possible executions, and

ii) a regulatory framework, describing the compliance requirements.

2.1 Structured Business Processes

Generally, process models can be seen as a compact way to represent the set of possible ways that a company have to achieve some given business objectives. These models contain the tasks, which correspond to the atomic activities that can be executed to bring forward the achievement of the business objective pursued by the executions included in the model. These tasks are organised within the process model and describe a set of possible orders in which they can be executed. Example 1 illustrates an instance of a process that can be possibly used to describe the sale procedures in a shop.

Example 1 (Shop Sale Process)**.** *Considering the scenario of a shop selling goods to costumers, the sale procedure can be summarised as a process by considering the sequence of steps listed below:*

1. *The customer chooses the goods he/she wants to purchase.*
2. *The total cost of the goods is tallied up.*
3. *The customer pays the calculated amount.*
4. *The sale is concluded.*

Using such formal models to represent their business procedures, companies allow to ensure that such procedures follow the required regulations by checking these models.

In this paper we focus our analysis on structured process models, such type of processes is similar to structured workflows defined by [44]. The advantage of focusing our initial analysis on these kind of processes is that their soundness[1] can be verified in polynomial time with respect to their size, and that the amount of possible executions belonging to the process model is finite, as it does not contain *loops* that can be potentially iterated any number of times, leading to business process models containing an infinite amount of possible executions.

Despite their simplicity, such kind of business process models can be used to represent 406 out of 604 processes in the SAP reference model [42], as shown by [55], illustrating also that unstructured processes, under certain conditions, can be translated into structured process models.

Definition 1 (Process Block)**.** *A process block B is a directed graph: the nodes are called* elements *and the directed edges are called* arcs*. The set of elements of a process block are identified by the function $V(B)$ and the set of arcs by the function $E(B)$. The set of elements is composed of tasks and coordinators. There are 4 types of coordinators:* and_split*,* and_join*,* xor_split *and* xor_join*. Each process block B has two distinguished nodes called the* initial *and* final *element. The initial element has no incoming arc from other elements in B and is denoted by $b(B)$. Similarly the final element has no outgoing arcs to other elements in B and is denoted by $f(B)$.*

A directed graph composing a process block is defined inductively as follows:

- *A single task constitutes a process block. The task is both initial and final element of the block.*

[1] A process is sound, as defined by van der Aalst [67, 68], if it avoids livelocks and deadlocks.

- *Let* $B_1, \ldots, B_n$ *be distinct process blocks with* $n > 1$*:*
  - $\mathsf{SEQ}(B_1, \ldots, B_n)$ *denotes the process block with node set* $\bigcup_{i=0}^{n} V(B_i)$ *and edge set* $\bigcup_{i=0}^{n}(E(B_i) \cup \{(f(B_i), b(B_{i+1})) : 1 \leq i < n\})$.
    The initial element of $\mathsf{SEQ}(B_1, \ldots, B_n)$ *is* $b(B_1)$ *and its final element is* $f(B_n)$.
  - $\mathsf{XOR}(B_1, \ldots, B_n)$ *denotes the block with vertex set* $\bigcup_{i=0}^{n} V(B_i) \cup \{\mathsf{xsplit}, \mathsf{xjoin}\}$ *and edge set* $\bigcup_{i=0}^{n}(E(B_i) \cup \{(\mathsf{xsplit}, b(B_i)), (f(B_i), \mathsf{xjoin}) : 1 \leq i \leq n\})$ *where* xsplit *and* xjoin *respectively denote an* xor_split *coordinator and an* xor_join *coordinator, respectively. The initial element of* $\mathsf{XOR}(B_1, \ldots, B_n)$ *is* xsplit *and its final element is* xjoin.
  - $\mathsf{AND}(B_1, \ldots, B_n)$ *denotes the block with vertex set* $\bigcup_{i=0}^{n} V(B_i) \cup \{\mathsf{asplit}, \mathsf{ajoin}\}$ *and edge set* $\bigcup_{i=0}^{n}(E(B_i) \cup \{(\mathsf{asplit}, b(B_i)), (f(B_i), \mathsf{ajoin}) : 1 \leq i \leq n\})$ *where* asplit *and* ajoin *denote an* and_split *and an* and_join *coordinator, respectively. The initial element of* $\mathsf{AND}(B_1, \ldots, B_n)$ *is* asplit *and its final element is* ajoin.

By enclosing a process block as defined in Definition 1 along with a start and end task in a sequence block, we obtain a *structured process model*. Therefore, a structured process model can be understood as a structure recursively composed by process blocks, where at the lowest recursion level are the process blocks representing the tasks of the process model.

The effects of executing the tasks contained in a business process model are described using annotations as shown in Definition 2.

Definition 2 (Annotated process)**.** *Let* P *be a structured process and* T *be the set of tasks contained in* P*. An annotated process is a pair:* (P, ann)*, where* ann *is a function associating a consistent set of literals to each task in* T*:* $\mathsf{ann} : T \mapsto 2^{\mathcal{L}}$.

The status of the process execution is represented by a process' state. Such state contains a set of literals representing what is considered to be the case at that step of the execution. The literals contained in the process' state is determined by the sequence of the task being executed, and it is updated after each task execution.

The update between the states of a process during its execution is inspired by the AGM[2] belief revision operator [2] and is used in the context of business processes to define the transitions between states [23, 39], which in turn are used to define the *traces*.

[2]The approach is named after the initials of the authors introducing it: Alchourrón, Gärdenfors, and Makinson.

Definition 3 (State update)**.** *Given two consistent sets of literals L_1 and L_2, representing the process state and the annotation of a task being executed, the update of L_1 with L_2, denoted by $L_1 \oplus L_2$ is a set of literals defined as follows:*

$$L_1 \oplus L_2 = L_1 \setminus \{\neg l \mid l \in L_2\} \cup L_2$$

Definition 4 (Executions and Traces)**.** *Given a structured process model identified by a process block B, the set of its executions, written $\Sigma(B) = \{\epsilon | \epsilon$ is a sequence and is an execution of $B\}$. The executions contained in $\Sigma(B)$ are recursively constructed as follows:*

1. *If B is a task t, then $\Sigma(B) = \{(t)\}$*
2. *if B is a composite block with sub-blocks $B_1, \ldots, B_n$:*
   (a) *If $B = \mathsf{SEQ}(B_1, \ldots, B_n)$, then $\Sigma(B) = \{\epsilon_1 +_{\mathcal{E}} \cdots +_{\mathcal{E}} \epsilon_n | \epsilon_i \in \Sigma(B_i)\}$ and $+_{\mathcal{E}}$ the operator concatenating two executions.*
   (b) *If $B = \mathsf{XOR}(B_1, \ldots, B_n)$, then $\Sigma(B) = \Sigma(B_1) \cup \cdots \cup \Sigma(B_n)$*
   (c) *If $B = \mathsf{AND}(B_1, \ldots, B_n)$, then $\Sigma(B) = \{$the union of the interleavings of: $\epsilon_1, \ldots, \epsilon_n | \epsilon_i \in \Sigma(B_i)\}$*

Given an annotated process (B, ann) and an execution $\epsilon = (t_1, \ldots, t_n)$ such that $\epsilon \in \Sigma(B)$, a trace θ is a finite sequence of states: $(\sigma_1, \ldots, \sigma_n)$. Each state $\sigma_i \in \theta$ is a pair: (t_i, L_i) capturing what holds after the execution of a task t_i, expressed by a set of literals L_i. A set L_i is constructed as follows:

1. *$L_0 = \emptyset$*
2. *$L_{i+1} = L_i \oplus \mathsf{ann}(t_{i+1})$, for $1 \leq i < n$.*

To denote the set of possible traces resulting from a process model (B, ann), we use $\Theta(B, \mathsf{ann})$.

Example 2. *Annotated Process Model. Fig. 1 shows a structured process containing four tasks labelled t_1, t_2, t_3 and t_4 and their annotations. The process contains an* AND *block followed by a task and an* XOR *block nested within the* AND *block. The annotations indicate what has to hold after a task is executed. If t_1 is executed, then the literal a has to hold in that state of the process.*

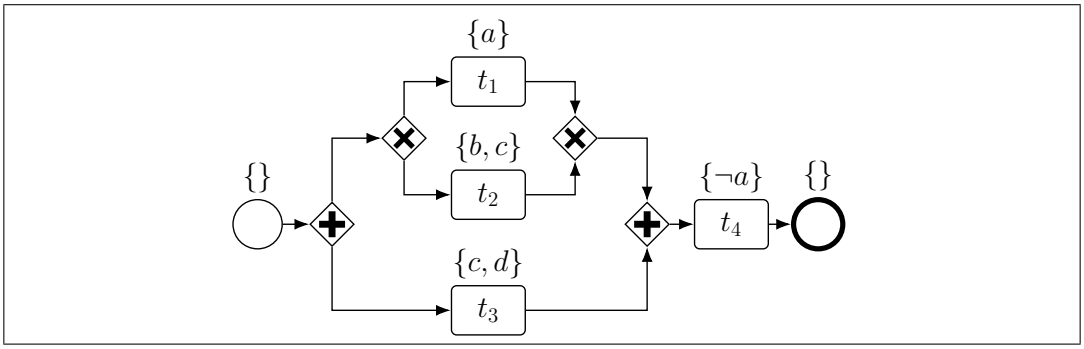


Figure 1: An annotated process

$\Sigma(B)$	$\Theta(B, \mathsf{ann})$
$(\mathsf{start}, t_1, t_3, t_4, \mathsf{end})$	$((\mathsf{start}, \emptyset), (t_1, \{a\}), (t_3, \{a, c, d\}), (t_4, \{\neg a, c, d\}), (\mathsf{end}, \{\neg a, c, d\}))$
$(\mathsf{start}, t_2, t_3, t_4\mathsf{end})$	$((\mathsf{start}, \emptyset), (t_2, \{b, c\}), (t_3, \{b, c, d\}), (t_4, \{\neg a, b, c, d\}), (\mathsf{end}, \{\neg a, b, c, d\}))$
$(\mathsf{start}, t_3, t_1, t_4\mathsf{end})$	$((\mathsf{start}, \emptyset), (t_3, \{c, d\}), (t_1, \{a, c, d\}), (t_4, \{\neg a, c, d\}), (\mathsf{end}, \{\neg a, c, d\}))$
$(\mathsf{start}, t_3, t_2, t_4\mathsf{end})$	$((\mathsf{start}, \emptyset), (t_3, \{c, d\}), (t_2, \{b, c, d\}), (t_4.\{\neg a, b, c, d\}), (\mathsf{end}, \{\neg a, b, c, d\}))$

Table 1: Executions and Traces of the annotated process in Fig. 1.

2.2 Regulatory Framework

When considering a compliance problem, one of its components is the set of regulations that the model is required to follow. We refer to such component as the regulatory framework, and we consider it as a set obligations, representing the set of regulations governing the process model within the scope of the problem.

As such, we use a subset of Process Compliance Logic (PCL), introduced by Governatori and Rotolo [30, 31], to specify the semantics for different types of obligations proposed by [38].

The first distinction in the semantics of obligations, is that obligations can be either *global* or *local*. A global obligation is in force for the entire duration of an execution, while the in force interval of a local obligation is determined by its trigger and deadline conditions. Additionally, an obligation can be either an achievement or a maintenance obligation and it determines how such an obligation is fulfilled by an execution when in force.

Definition 5 (Global and Local Obligations)**.** *The in force interval of an obligation depends on whether it is a global or a local obligation, described as follows:*

Global *A global obligation $\mathcal{O}^o\langle c\rangle$, where $o \in \{a, m\}$ represents whether the obligation is achievement or maintenance. The element c represents the fulfilment condition of the obligation.*

Local *A local obligation $\mathcal{O}^o\langle c, t, d\rangle$, where $o \in \{a, m\}$ represents whether the obligation is achievement or maintenance. The element c represents the fulfilment condition of the obligation, the element t the trigger, and the element d the deadline.*

While the in force interval of a global obligation spans the entire duration of a trace, the in force interval of a local obligation is determined as a sub-trace where the first state of such a sub-trace satisfies the trigger, and the last state satisfies the deadline.

Generally the trigger, deadline and condition of an obligation are defined as propositional formulae. Assuming the literals from $\mathcal{L}$ contained in a state to be true, then a propositional formula is true when that state implies it.

Finally, in the semantic we study for each obligation we allow a single in force interval at any given time. Meaning that when an in force interval is already active for an obligation, further triggers would not produce additional in force intervals. This has the consequence that it simplifies the analysis as it is not required to keep track of multiple in force instances, and which in force instance is satisfied by which event when executing a task.

Evaluating the Obligations.

Whether an in force obligation is fulfilled or violated is determined by the states of the trace included in the in force interval of the obligation. Moreover, whether an in force obligation is fulfilled depends on the type of an obligation, as described in Definition 6.

Definition 6 (Achievement and Maintenance Obligations)**.** *How an in force obligation is fulfilled depends on its type as follows:*

Achievement *If this type of obligation is in force in an interval, then the fulfilment condition specified by the obligation must be satisfied by the execution in at least one point in the interval before the deadline is satisfied. If this is the case, then the obligation in force is considered to be satisfied. Otherwise it is violated.*

Maintenance *If this type of obligation is in force in an interval, then the fulfilment condition must be satisfied* continuously *in all points of the interval until the deadline is satisfied. Again, if this is the case, then the obligation in force is then satisfied, otherwise it is violated.*

Process Compliance.

The procedure of proving whether a process is compliant with a regulatory framework can return different answers. A process is said to be *fully compliant* if every trace of the process is compliant with the regulatory framework[3]. A process is *partially compliant* if there exists at least one trace that is compliant with the regulatory framework, and *n*ot compliant if there is no trace complying with the framework.

Definition 7 (Process Compliance)**.** *Given a process* (P, ann) *and a regulatory framework composed by a set of obligations* $\mathbb{O}$*, the compliance of* (P, ann) *with respect to* $\mathbb{O}$ *is determined as follows:*

- ***Full compliance*** $(P, \mathsf{ann}) \vdash^{F} \mathbb{O}$ *if and only if* $\forall \theta \in \Theta(P, \mathsf{ann})$*,* θ *satisfies each obligation in* $\mathbb{O}$*.*
- ***Partial compliance*** $(P, \mathsf{ann}) \vdash^{P} \mathbb{O}$ *if and only if* $\exists \theta \in \Theta(P, \mathsf{ann})$*,* θ *satisfies each obligation in* $\mathbb{O}$*.*
- ***Not compliant*** $(P, \mathsf{ann}) \nvdash \mathbb{O}$ *if and only if* $\neg\exists \theta \in \Theta(P, \mathsf{ann})$*,* θ *satisfies each obligation in* $\mathbb{O}$*.*

Note that we consider a trace to be compliant with a regulatory framework if it satisfies every obligation belonging to the set composing the framework.

3 Theoretical Computational Complexity in Structured Process Models

In this section we discuss the existing results concerning verifying regulatory compliance of structured business process models. We first introduce the acronyms used through the section to identify the different variants of the problem, and then we separately analyse and discuss the computational complexity results related to full, and partial compliance separately.

3.1 Problem Acronyms

Before discussing the existing computational complexity results, we first introduce a compact system to refer to different variants of the problem dealing with verifying compliance of structured process models. Notice that the acronyms refer to

[3]Notice that by "compliant with the regulatory framework", we refer to a trace fulfilling each in force interval along the trace itself for each obligation belonging to the regulatory framework.

the properties of the regulatory framework being evaluated against the structured process model.

Definition 8 (Compact Acronyms)**.** *The variants of the problem we refer to in this paper constantly aim to check regulatory compliance of a structured process model. The acronym system refers to the properties of the obligations being checked against the process model.*

1/n *Whether the structured process is checked against a single (**1**) or a set of (**n**) obligations.*

G/L *Whether the* in force *interval of the obligations is **G**lobal, meaning that it spans the entirety of an execution of the model, or it is **L**ocal, meaning that the in force interval is determined by the trigger and deadline elements of an obligation.*

-/+ *Whether the elements of the obligation being checked on the structured process model are composed by literals (-), or by propositional formulae (+).*

For instance, the variant **1G-** consists of verifying whether a structured process model is compliant with a single obligation, whose condition is expressed as a propositional literal and its in force interval spans the entire execution of the process model.

Note that in the binary properties of the problems considered in this paper, the leftmost, i.e., **1** in **1/n** represents a subset of the right side. Intuitively, the case on the right side is at least as complex as the left case. For instance, a solution for a problem including a set of regulations requires also to solve the case where the set of regulations is composed of exactly one regulation.

3.2 Partial Compliance

We focus now on discussing the computational complexity of proving partial compliance of structured business process models. As we see in the remainder of this section, many of the variants belong to the **NP**-complete computational complexity class. Thus we provide quick reminder before proceeding by discussing the existing results.

Definition 9 (NP-complete)**.** *A decision problem is **NP**-complete if it is in the set of **NP** problems and if every problem in **NP** is reducible to it in polynomial-time.*

To prove membership in **NP** of a variant of the problem of proving partial compliance, we show that a process is partially compliant with a set of obligations if and only if there is a certificate whose size is at most polynomial in terms of the length of the input (comprising the business process model and the set of obligations) with which we can check whether it fulfils the regulatory framework in polynomial time. As a certificate we use a trace of the model and we check whether it satisfies the regulatory framework.

We illustrate in the following Algorithm 1 how **1G-** is solvable in time polynomial. Notice that, while the algorithm reported applies only to achievement obligations, in the original paper by Colombo Tosatto et al. [14], from which we took this approach, an algorithm dealing with a regulatory framework composed of a maintenance obligation is also provided. Moreover, notice that the algorithm reported is capable to prove either partial, full, and non-compliance in polynomial time.

Algorithm 1 (**1G-** is in **P**)**.** *Given an annotated process* (P, ann) *and a regulatory framework* $\mathbb{O}$ *containing a single global achievement obligation* $\mathcal{O}^a\langle c\rangle$*, this algorithm returns whether* (P, ann) *is compliant with* $\mathbb{O}$*.*

1: ***if*** $\forall$ *task* t *in* $P, c \notin \mathsf{ann}(t)$ ***then***
2: ***return*** $(P, \mathsf{ann}) \not\vdash \mathbb{O}$
3: ***else***
4: ***if*** $\mathsf{Remove}(P, \{t \mid t \text{ is a task in } P \text{ and } c \in \mathsf{ann}(t)\}) = \bot$ ***then***
5: ***return*** $(P, \mathsf{ann}) \vdash^F \mathbb{O}$
6: ***else***
7: ***return*** $(P, \mathsf{ann}) \vdash^P \mathbb{O}$
8: ***end if***
9: ***end if***

Where the Remove *functions removes the tasks from* P *having* c *annotated, and later checks whether there is a path, in other words an execution, from the start to the end of the process. If no such path exists then the function returns* $\bot$*, which means that there is no execution that does not execute a task having* c *annotated. Meaning that the process is fully compliant.*

In Reduction 1 we show the reduction provided by Colombo Tosatto et al. [10], and showing that the problem of finding whether a graph contains an Hamiltonian path can be reduced to the problem of proving partial compliance in **nL-**. Meaning that the computational complexity of **nL-** is at least the same as proving whether a graph contains an Hamiltonian path, which is in **NP**-complete.

Definition 10 (Hamiltonian Path)**.** *Let* $G = (N, D)$ *be a directed graph where the size of* N *is* n*. A hamiltonian path* $ham = (v_1; \ldots; v_n)$ *satisfies the following*

properties:

1. $N = \{v_1, \ldots, v_n\}$
2. $\forall i, j((v_i, v_j \in ham \wedge j = i + 1), ((v_i, v_j) \in D)$

Reduction 1 (Hamiltonian Path to Proving Partial Compliance in **nL-**)**.** *Considering the problem of finding an Hamiltonian Path in a graph as described in Definition 10.*

Given a hamiltonian path problem containing a directed graph $G = (N, D)$, it can be translated into a regulatory compliance problem involving a process (P, ann) and a set of obligations $\mathbb{O}$ as follows:

1 *Consider a process model P that contains a task labeled $Node_i$ for each vertex v_i contained in N (Figure 2).*

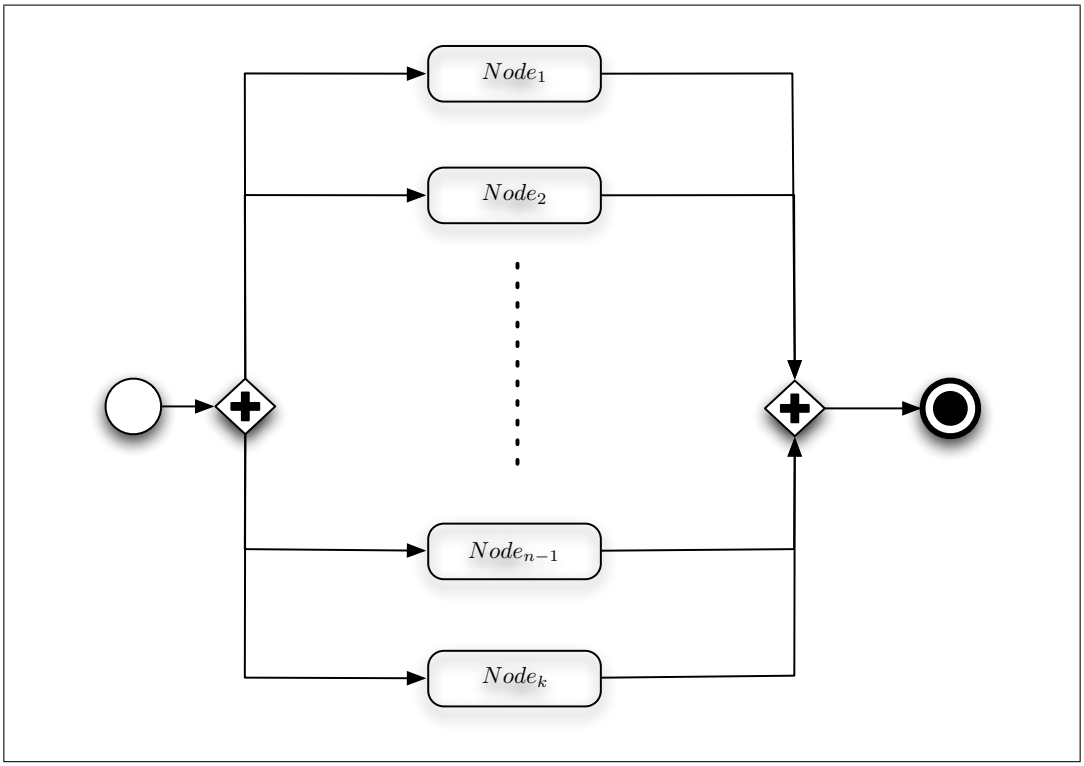


Figure 2: Hamiltonian path problem as verifying partial compliance.

The process block of P is structured as an AND *block. The* AND *block contains in each branch a single task $Node_i$ for each node in the given directed graph:* $\mathsf{AND}(Node_1, \ldots, Node_n)$.

Intuitively a serialisation of the AND *block represents a tentative hamiltonian path. Annotations and obligations are used to verify that two adjacent nodes in the serialisation can be indeed also adjacent in an hamiltonian path (explained in detail in* ***2****).*

2 *In this reduction we use the annotations to identify which node is being selected in the sequence constituting the tentative hamiltonian path. Thus we use for the annotations a language containing a literal for each node in G. The annotation of each task in (P, ann) is the following:*

- $\forall i | 1 \leq i \leq k, \mathsf{ann}(Node_i) = \{\neg l_1, \ldots, \neg l_n\} \oplus \{l_i\}$

The obligations are used to represent the directed edges departing from a vertex, in other words which vertices are the suitable successors in the hamiltonian path. The set $\mathbb{O}$ contains the following local maintenance obligations:

- $\forall v_i, v_j | (v_i, v_j) \notin D, \mathcal{O} = \mathcal{O}^m \langle \neg l_j, l_i, \neg l_i \rangle$

Using the proposed reduction, verifying whether the constructed process model is partially compliant corresponds to identifying whether the original graph contains a hamiltonian path. Concluding that the problem of verifying partial compliance is at least as hard as finding whether a graph contains a hamiltonian path.

We collect in Table 2 the existing computational complexity results concerning solving the variants of the problem of proving partial compliance of structured process models.

Notice that given the three binary properties associated to the regulatory framework being checked against the structured process model, of the 8 possible problem variants, only 7 computational complexity results are provided in Table 2. This is more apparent by illustrating the results in Figure 3, where the problem's variants have their computational complexities associated and the relations between the variants are highlighted by the connections in the picture. Notice that the directed arrows connecting one variant of the problem to another refer, according to their direction, that the computational complexity of a variant of the problem at the origin of an arrow, is at most as difficult as the variant of the problem which is pointed at by the same arrow.

Problem Variant	Source	Complexity Class
1G-	Colombo Tosatto et al. [14]	**P**
nG-	Colombo Tosatto et al. [11]	**NP**-complete
1G+	Colombo Tosatto et al. [11]	**NP**-complete
nG+	Colombo Tosatto et al. [11]	**NP**-complete
nL-	Colombo Tosatto et al. [10]	**NP**-complete
1L+	Colombo Tosatto et al. [11]	**NP**-complete
nL+	Colombo Tosatto et al. [10]	**NP**-complete

Table 2: Partial Compliance Complexity

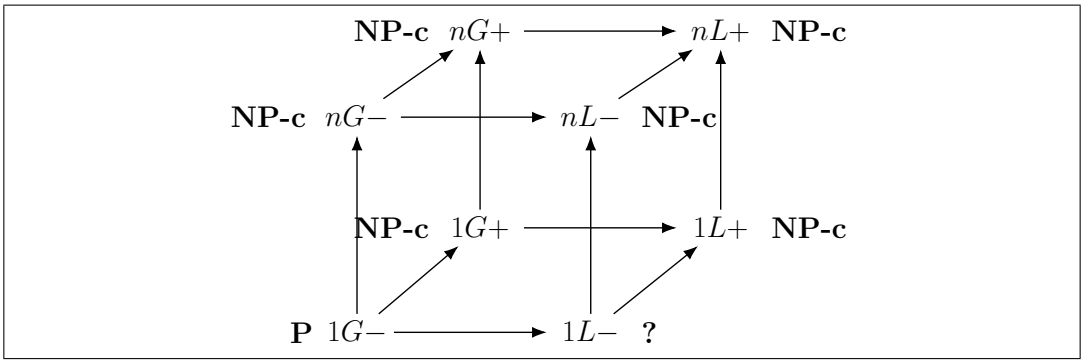


Figure 3: Partial Compliance Complexity Lattice.

It can be noticed in Figure 3, that the variant **1L-** does not have a computational complexity classification yet. While the computational complexity analysis for the considered problem is currently incomplete, Colombo Tosatto et al. [11] conjectured that this variant of the problem to in **P**. However, while we have not yet managed to provide a conclusive computational complexity classification, we conjecture that **1L-** is instead in **NP-c** as explained in Conjecture 1

Conjecture 1 (**1L-** is in **NP**)**.** *We currently have no information about the computational complexity of **1L-**. That is, we cannot infer its belonging to a computational complexity class in a similar way as for **nG+**, as in this case the simpler variant (**1G-**) is in **P**.*

*While it seems like that moving from **G** to **L** seems to not increase the complexity of the problem as much as when moving from **-** to **+**, or from **1** to **n**, we believe that such movement should be still be capable of bringing the computational complexity of the problem's variant into **NP-c**.*

We back our conjecture using the intuition that by moving towards conditional

*obligations, allows multiple instances of the same obligation to be in force over a single trace. Which means that even for the variant **1L-**, multiple instances would be required to be verified for every trace. Which resembles the variant **nG-**, where multiple obligations are required to be verified over a trace, and it is in **NP-c**.*

Mind that the conjecture does not represent a computational complexity result in itself, hence identifying the computational complexity of the variant **1L-** remains an open problem.

3.3 Full Compliance

We focus now on discussing the computational complexity of the variants of the problem of proving full compliance of a structured process model. As many of the variants of the problem belong to the co**NP**-complete computational complexity class, we provide its definition before proceeding with the discussion.

Definition 11 (co**NP**-complete)**.** *A decision problem is co**NP**-complete if it is in co**NP** and if every problem in co**NP** is polynomial-time many-one reducible to it. A decision problem is in co**NP** if and only if its complement is in the complexity class **NP**.*

We show in Reduction 2 how Colombo Tosatto et al. [10] have shown that checking for full compliance in a variant of the problem **1L+** is in co**NP**-complete.

Definition 12 (Tautology)**.** *A formula of propositional logic is a tautology if the formula itself is always true regardless of which evaluation is used for the propositional variables.*

Reduction 2 (Tautology to Proving Full Compliance in **1L+**)**.** *Considering the problem to decide whether a given formula is a Tautology as described in Definition 12.*

Let φ be a propositional formula for which we want to verify whether it is a tautology or not, and let L be the set of literals contained in φ. We include in L only the positive version of a literal, for instance if l or $\neg l$ are contained in φ, then only l is included in L.

For each literal l belonging to L we construct an XOR *block containing two tasks, one labeled and containing in its annotation the positive literal (i.e., l) and the other the negative literal (i.e., $\neg l$). All the* XOR *blocks constructed from L are then included within a single* AND *block. This* AND *block is in turn followed by a task labeled "test" and containing a single literal in its annotation: l_{test}. The sequence containing the* AND *block and the task* test *is then enclosed within a* start *and an* end*, composing the process (P, ann), graphically represented in Figure 4.*

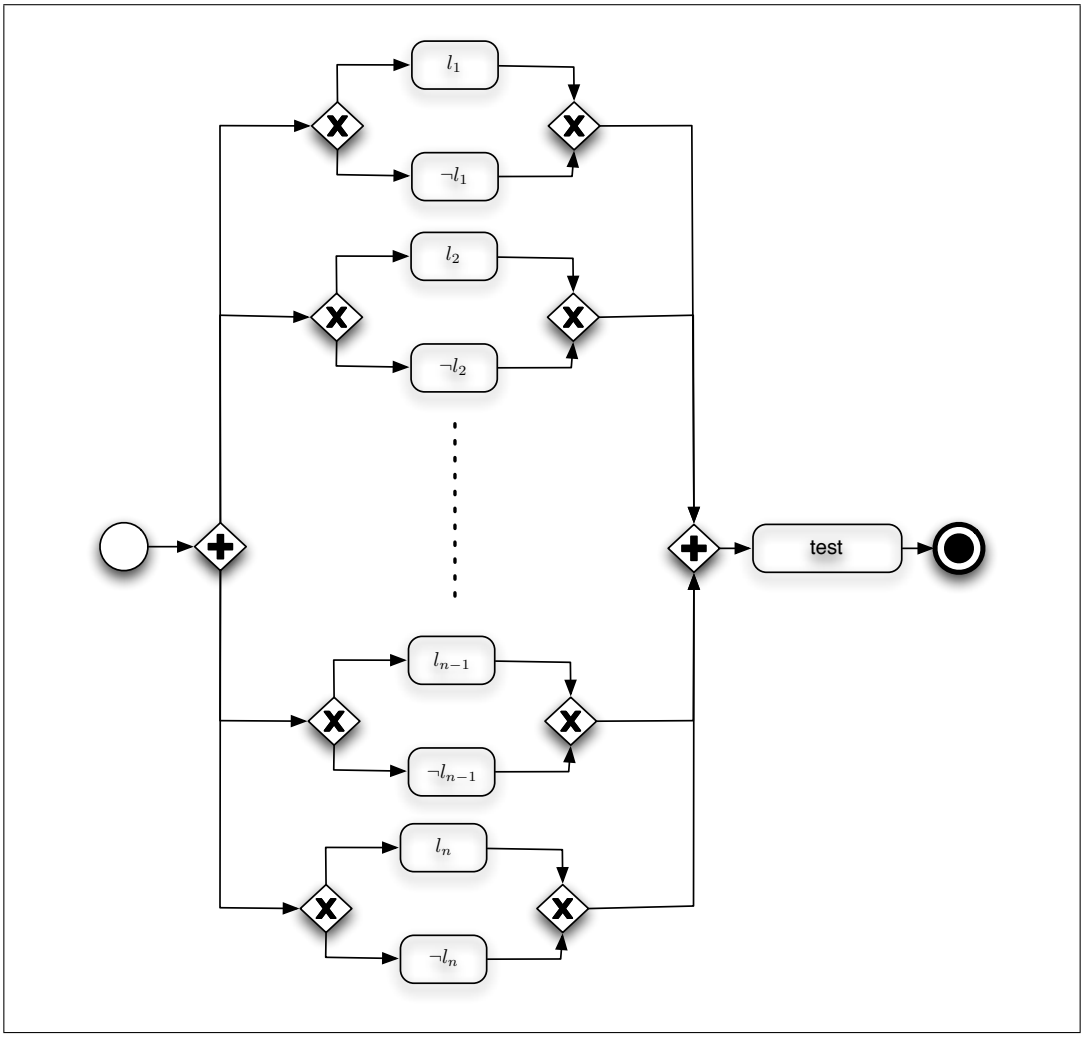


Figure 4: Tautology problem as verifying full compliance.

The set of obligations, to which the constructed process has to be verified to be fully compliant with, is composed of a single obligation constructed as follows from the propositional formula φ:

$$\mathcal{O}^a\langle \varphi, l_{test}, \bot \rangle$$

Notice that Algorithm 1 can also be used to prove full compliance of the variant **1G-**. For full compliance, we informally describe in Algorithm 2 the procedure

introduced by Colombo Tosatto et al. [12], capable of proving full compliance of the variant **nL-** of the problem in time polynomial with respect to the problem size.

Definition 13 (Process Tree Model)**.** *Let P be a structured process model. A* Process Tree *PT is an abstract hierarchical representation of P, where:*

- *Each process block B in P corresponds to a node N in PT.*
- *Given a process block $B(B_1, \ldots, B_n)$, where $B_1, \ldots, B_n$ are the process blocks directly nested in B, the nodes $N_1, \ldots, N_n$ in PT, corresponding to $B_1, \ldots, B_n$ in P, are children of N, corresponding to B in P. Mind that the order between the sub-blocks of a process block is preserved between the children of the same node.*

Algorithm 2. *The approach is based on identifying whether a structured business process model, in its tree representation form as described in Definition 13, contain a trace violating one of the obligations belonging to the regulatory framework.*

The advantage of looking for a violating condition, is that finding a single instance within a process model where such condition is positively evaluated, it is a sufficient condition to return the answer that the structured process being evaluated is not fully compliant *with the regulatory framework.*

The tree representation of a process model has as its leaves the tasks composing the process. Considering a generic obligation $\mathcal{O}^o\langle c, t, d\rangle$, each of the leaves in a process tree associated to a task having t annotated are considered trigger leaves*. A bottom up aggregation of the properties of the leaves of the tree, in accordance to their associated annotated tasks, and to the violation condition being looked for, leads to allow whether a process tree contains a violation for a given* trigger leaf *in a number of steps equal to the number of nodes in the process tree.*

Repeating this procedure for each trigger leaf, for each violation condition of each obligation in a regulatory framework, allows to decide, when no violation condition is satisfied, that the business process model being evaluated is fully compliant, and this is decidable in time polynomial with respect to the size of the problem.

Table 3 outlines the existing complexity results concerning some of the variants of the problem of proving full compliance of structured process models.

Similarly as for partial compliance, we illustrate the result concerning the computational complexity of proving full compliance of a process model graphically in Figure 5.

Notice that Figure 5 contains a result for the problem variant **nG-**, which is not included in Table 3. This result is derived from other existing ones. As the relations

Problem Variant	Source	Complexity Class
1G-	Colombo Tosatto et al. [14]	**P**
1L-	Colombo Tosatto et al. [12]	**P**
nL-	Colombo Tosatto et al. [12]	**P**
1G+	Colombo Tosatto et al. [13]	co**NP**-complete
nG+	Colombo Tosatto et al. [13]	co**NP**-complete
1L+	Colombo Tosatto et al. [10]	co**NP**-complete
nL+	Colombo Tosatto [9]	co**NP**-complete

Table 3: Full Compliance Complexity

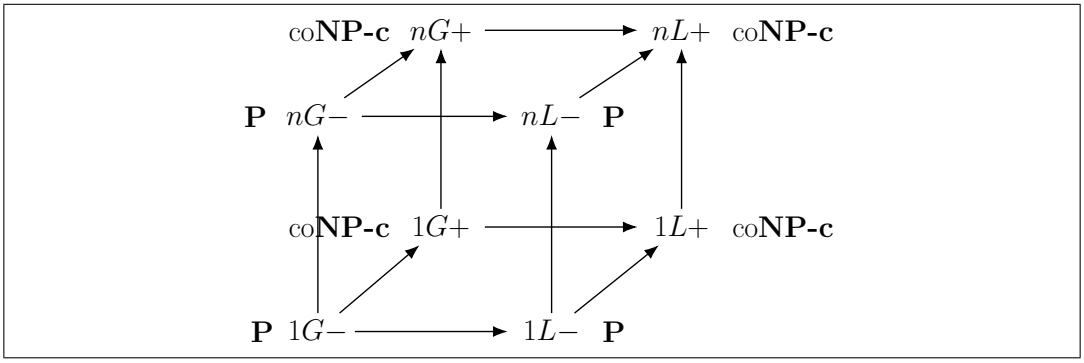


Figure 5: Full Compliance Complexity Lattice.

in the lattice in Figure 5 represent the relation $\leq$ between the computational complexities of the variants of the problem, if we consider the three variants **1G-**, **nL-**, and **nG-**, the follow relationship holds regarding their computational complexity: **1G-** $\leq$ **nG-** $\leq$ **nL-**. Therefore, by knowing that both **1G-** and **nL-** are in **P**, it follows that also **nG-** is in **P**.

3.4 Climbing the Polynomial Hierarchy

The computational complexity results reported so far concerning the problems of proving both partial and full compliance of structured process model, rely on an assumption regarding how formulae composing the obligations are evaluated over the states composing the traces.

Assumption 1 (States Satisfying Formula)**.** *Given a state σ, composed by a set of positive and/or negative literals, and a propositional formula φ, φ is satisfied by σ if and only if considering every literal in σ true, is a sufficient interpretation to*

make φ true. This is equivalent to evaluate a formula over a partial set of possible interpretations, the ones that explicitly appear in the state.

While the assumption does not appear to be too surprising, it can lead to some interesting behaviours, such as the following: consider the formula $\alpha \vee \neg\alpha$, which is a tautology. Now, if we consider whether an empty state of a trace would satisfy, the formula, then the answer is counter-intuitively *no* in accordance to Assumption 1.

The effect of Assumption 1 on how formulae composing the obligations are verified over the states of the traces, is to simplify the verification, as the only interpretation required to be verified is the one where every proposition in the state is considered as true. If such interpretation is sufficient to evaluate the formula as true, then its associated effects, according to its place in the obligation, are applied. Differently, if the interpretation provided by the state is not sufficient to fully evaluate the formula, then it is assumed to be false in that state. Normally, without Assumption 1, when a state does not contain a sufficient interpretation, then the various cases are considered for the propositions which have not an assigned truth value. This can potentially increase the computational complexity of solving the problem, as evaluating a formula over a state can be reduced to a *Satisfiability* problem, which is known to be **NP**-complete. In order to properly classify the variants of the problem when Assumption 1 is dropped, we need first to introduce the *Polynomial Hierarchy.*

The Polynomial Hierarchy is a hierarchy of computational complexity classes describing both the classes already discussed in the present chapter (**P**, **NP** and co**NP**), as well as more complex classes. In the Polynomial Hierarchy **P** is also represented as either Σ_0^{P} or Π_0^{P}, while **NP** and co**NP** are respectively represented as Σ_1^{P} and Π_1^{P}.

Definition 14 (Σ_1^{P})**.** *A problem P is in Σ_1^{P} if there exists a polynomial time Turing machine T and a polynomial p such that:*

for each instance x of P: there exists a solution s, $|s| \leq p(|x|), T(x, s) = true$

Definition 15 (Π_1^{P})**.** *A problem P is in Π_1^{P} if there exists a polynomial time Turing machine T and a polynomial p such that:*

for each instance x of P: for each solution s, $|s| \leq p(|x|), T(x, s) = true$

Considering now the problem of proving partial compliance of a structured process model, when Assumption 1 is dropped. We have that for the variants of the problem allowing formulae to describe the elements of the obligations, the problem becomes the following: there exists a trace of the model such that, for each state

state in the trace, and for each possible interpretations of the state the formulae composing the obligations are satisfied in such a way that no obligation is violated. It can be noticed, that this problem involves an existential quantifier followed by two universal quantifiers: $\exists\forall\forall$. While not delving too much into the technical details, when multiple quantifier of the same type directly follow each other, they can be collapsed as a single quantifier of the same type. Given that, and Definition 16, we can see that the variants of the problem of proving partial compliance, and allow formulae in their obligations, can be classified as Σ_2^{P}. Furthermore, notice that the variants restricting the expressivity of their obligations to simple propositions are not affected and remain in Σ_1^{P}.

Definition 16 (Σ_2^{P})**.** *A problem P is in Σ_2^{P} if there exists a polynomial time Turing machine T and a polynomial p such that:*

for each instance x of P: there exists a solution s, $|s| \leq p(|x|)$ such that each s', $|s'| \leq p(|x|)$, $T(x, s, s') = true$

The lattice with the computational complexity classifications according to the Polynomial Hierarchy, after dropping Assumption 1, is shown in Figure 6.

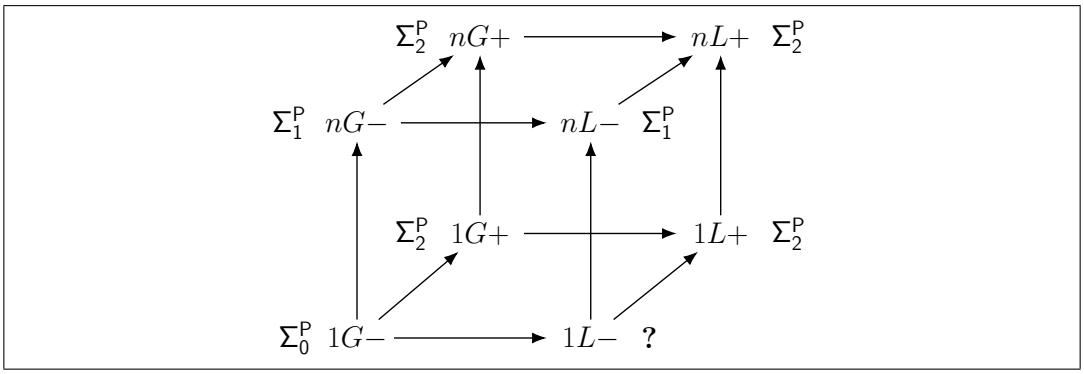


Figure 6: Partial Compliance Complexity Lattice with Polynomial Hierarchies.

Similarly, we can consider the problem of proving full compliance of structured process models, when Assumption 1 is dropped. Again, we have that for the variants allowing formulae in their obligations the problem becomes: for each trace of the model such that, for each state state in the trace, and for each possible interpretations of the state the formulae composing the obligations are satisfied in such a way that no obligation is violated. It can be noticed, that this problem involves three universal quantifiers: $\forall\forall\forall$. Again, we can collapse the quantifiers of the same type with the neighbouring ones, which leads to a single universal quantifier in this case. It can be noticed that these problems do not have the sufficient properties to be

classified as Π_2^P, as described in Definition 17, but they can be classified as Π_1^P, as described in Definition 15.

Definition 17 (Π_2^P)**.** *A problem P is in Π_2^P if there exists a polynomial time Turing machine T and a polynomial p such that:*

for each instance x of P: for each solution s, $|s| \leq p(|x|)$ such that there exists a s', $|s'| \leq p(|x|)$, $T(x, s, s') = true$

Therefore we can conclude that for the problem of proving full compliance, releasing Assumption 1 does not increase the complexity, as the variants allowing propositional formulae are still in co**NP**. For completeness we show the lattice with the Polynomial Hierarchy classifications for the variants of the problem of proving full compliance if Figure 7.

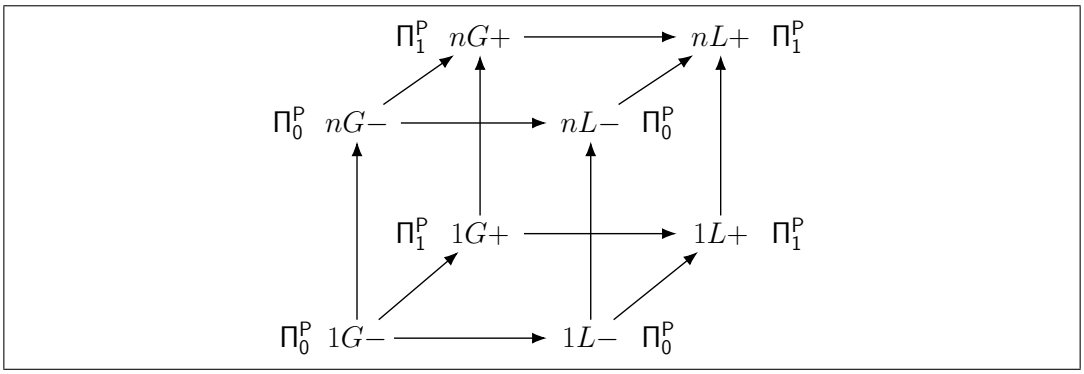


Figure 7: Full Compliance Complexity Lattice with Polynomial Hierarchies.

3.5 Summary

The computational complexity results illustrated in this section show that, when considering variants of the problem only allowing literals to represent the components of the obligations, proving full compliance of structured process model is generally easier than proving partial compliance. Intuitively, the former is easier since it is sufficient to find a violation for an obligation in one of the traces of the model, while for the former, it is required to identify a trace, and ensure that no obligation is violated in such a trace.

Moreover, it can be noticed from the computational complexity lattice in Figure 7, that for full compliance the computational complexity is be governed by the complexity of the language, namely by how expressively we can represent the elements representing the obligations.

Differently, partial compliance seems to be more complex to verify, as it does not seem to allow an easy way to identify its complement, and identifying a compliant trace in a process model is shown to be intractable apart from the the easiest, or maybe two easiest, variant(s) of the problem as illustrated in Figure 6.

Finally, we have also shown that by dropping Assumption 1, the computational complexity of the problem of proving partial compliance starts to climb the Polynomial Hierarchy.

4 Computational Complexity of Additional Business Process Features

The variants of the problem discussed earlier in this chapter have their computational complexity depending solely on the varying properties of the regulatory framework while keeping the properties of the process model static. Additional computational complexity analysis can be done when considering more complex variants of the process models used.

In this section we discuss about the computational complexity of proving compliance of process models by including additional features in the process models. In particular we discuss about the computational complexity of verifying compliance of unstructured process model, and the computational complexity impact of including loops in business process models.

4.1 Unstructured Process Models

The computational complexity analysis included in Section 3 focused on problems where the the process models were structured. As mentioned earlier, one of the advantages of such models is that their soundness can be verified in time polynomial with respect to the size of the model. Verifying soundness means to check whether every execution in the model is a proper execution, and capable of reaching the *end* of the model. In other words checking that the process model do not contain livelocks and or deadlocks preventing any of the contained execution to successfully complete.

Unstructured business process models, which are not composed by properly nested process blocks, as the instance shown in Figure 8, do not guarantee that their soundness can be verified in polynomial time. As it has been shown by van der Aalst [66], and Lohmann et al. [47], the semantics of business process models can be captured by Petri Nets [52]. While this does not provide any direct result concerning the computational complexity about verifying compliance of unstructured process

model, it still provides an upper complexity bound, as coloured petri nets, one of the more complex variants of this formalism, is known to be undecidable [19, 51].

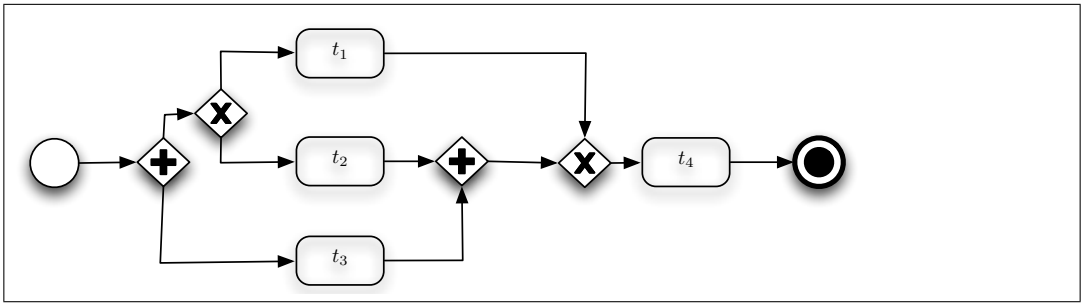


Figure 8: Unstructured Process Model

Open Problem

Studying the computational complexity of the problem of proving regulatory compliance of unstructured process models is still an open problem. Knowing already that if a unstructured process model is complex enough to require a coloured petri net to represent its semantics, then also the corresponding problem of verifying compliance becomes undecidable. Therefore, one option is to study which classes of unstructured processes do not require coloured petri nets to represent their semantics, and how they affect the computational complexity of solving the problem of proving regulatory compliance.

When studying how unstructured process models affect the computational complexity of the compliance problem, it is also crucial to consider that such models can be potentially translated into structured models, as discussed by Polyvyanyy et al. [54]. However the computational complexity cost of such translations should be still factored in the computational complexity of the problem, which still requires to be studied.

4.2 Loops

Loops are structures that allow to repeat the execution of parts of a process model. Figure 9 shows a process model containing a loop structure, in this model, the execution of the pair of tasks t_4 and t_5 can be executed any number of times between one and infinite. By allowing repeated executions of some of its elements, some issues can arise affecting the computational complexity of the problem of verifying regulatory compliance. In this section we discuss some of out intuitions behind these possible issues affecting the computational complexity of the problem.

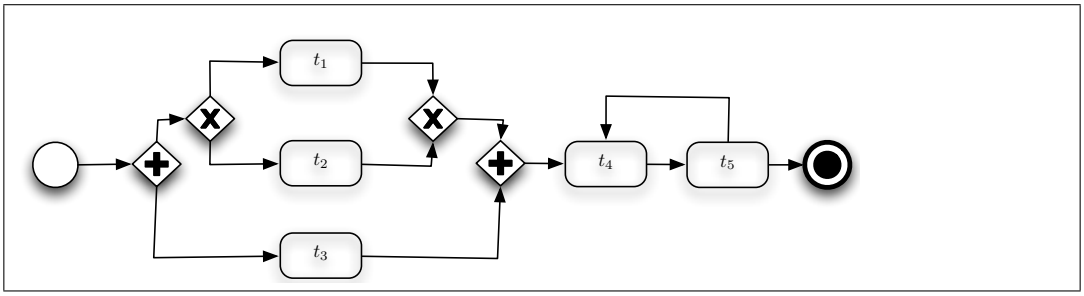


Figure 9: Process Model containing Loop

Infinite Possible Executions

By allowing repeated executions of some components of a process model, the maximal length of their executions is not constrained anymore by the number of tasks contained in the process model, but can become potentially infinite as a task within a loop can be executed an infinite number of times. Therefore, approaches based on verifying directly whether the traces[4] of a model comply with the regulatory framework, are not applicable to this variant of the problem as some of these traces can in theory be infinite, and such approaches could never terminate and return a result.

Therefore, introducing loops into the model would make pure brute force approaches, the ones analysing the generated traces of a model, completely obsolete. We argue that similar issues would apply to every approach analysing the traces instead of the structure of the model, however techniques to recognise repeating patterns in the process' states could be potentially used to recognise loops and study their compliance effects. However whether this techniques would work, and the actual computational complexity of the problem including loops are both unanswered research questions.

Structural Analysis

Differently, approaches that analyse the process's structure without the need to analyse each trace explicitly, would still be able to terminate in a finite amount of steps. Thus, we argue that such kind of approaches, which do not need to explicitly analyse the traces, are the only viable approaches when dealing with process models containing loops.

[4] As a reminder, a trace is the sequence of the tasks in an execution with associated process' states at every step of the execution.

However, the theoretical computational complexity of the variants of the problem identified in Section 3, with the inclusion of loops, is at least as difficult as the ones not including loops. This is due to the additional complexity brought by including the loops in the process model. While the added complexity may not necessarily be enough to increase the computational complexity class of the variants to harder ones, a through analysis of the computational complexity of the new variants including loops is still required in order to properly classify them.

Relating to Complete Turing Machines

We discuss now the intuition concerning why introducing loops in business process model, in addition to including conditions in the decision points[5], would allow potentially to simulate universal Turing machines using these extended business process models.

Definition 18 (Turing Completeness)**.** *In computability theory, a system of data-manipulation rules (such as a computer's instruction set, a programming language, or a cellular automaton) is said to be Turing complete or computationally universal if it can be used to simulate any Turing machine.*

Considering now adding into the business process models conditions for its loops, like starting and exit conditions, and decision condition for the mutual exclusive paths in the process model, it becomes more and more evident how the elements of a business process model can simulate different structures common to programming languages, such as various type of cycles and decision blocks. As these programming languages are generally known to be Turing complete languages, such as for instance *Java* and *C++*, considering the process state as the computational state of Turing machine, and the possible executions of a model as the possible computations of the Turing machine, then we can conclude that with these additions, such models become Turing complete.

As a consequence, considering the *halting problem*[6] [65] affecting Turing machines, it would then also affect the problem of proving regulatory compliance of business process models including loops. Which, in turn, would make the problem of proving compliance in general *undecidable*.

[5]We consider as decision points XOR blocks and loops, where a decision is required to be made concerning which branch to execute, or whether to exit the loop.

[6]The halting problem is the problem of determining, from a description of an arbitrary computer program and an input, whether the program will finish running or continue to run forever.

Relation to Fairness

Given the relation between the problem of proving regulatory compliance of business process models and computer programs, which becomes particularly clear when loops and conditions are introduced within the process model. It is only fair to discuss some of the related work dealing with the termination of computer programs, in particular as the computational complexity of dealing with such more advanced variants of the problem is related to the undecidability of halting problem.

The relation we are going to discuss is the one with the concept of *fairness for model checking verification* as discussed earlier by Francez [21], which can be intuitively understood as the following: given two properties of comparable importance for a problem, if a certain effort is put into the verification of one of the property, then it is only *fair* that the same amount of effort is put towards the verification of the other property. Such concept led to Cook et al. [15] to investigate the verification of liveness properties of programs in addition to their safety properties. Others, such as Dobrikov et al. [17] proposed implementations under *fairness* assumptions, capable of verifying how fairly model checking approaches perform, with limited overhead. Moreover, Kesten et al. [43] propose a fairness based approach based on LTL for model checking verification, showing that the introduction of fairness allows to close the gap with other approaches using CTL, as by using strong fairness, LTL properties can be verified on the model being checked without having to completely unfold the model to generate the possible states.

Adopting fairness to verify properties of models allows to do so without having to unfold these models and explicitly verify the possible states, in particular by adopting *strong fairness*, which is also referred as *compassion*. This requirement, compassion, stipulates that, given two types of states, then in every computation which verifies infinitely many of one of the types, is also required to verify infinitely many of the other type.

Considering now the problem of verifying regulatory compliance of business process models including loops, and assuming the existence of conditions governing choices such as mutually exclusive paths in the model, and whether a loop should be repeated or exited. It can be noticed that fairness based approached for model checking can be adapted to deal with such kind of problems. In particular, if we consider loops in a process model and its entering and exiting conditions as conditions leading to two different types of states for which we require *strong fairness*, then approaches verifying compliance, even by analysing the traces of the model explicitly, would be required to analyse an equivalent amount of states within and outside loops, guaranteeing to consider in such a way traces that represent full executions of the model. While this technique can prove useful to verify partial compliance

of process models, as it requires to find a compliant trace, further analysis may be necessary when dealing with full compliance, as in order to be classified as such, every possible trace must be considered and verified.

4.3 Summary

Despite the added expressivity introduced by using unstructured business process models and loops can be useful to represent real world problems more faithfully, how such additions would affect the computational complexity of the variants of the problem discussed in Section 3 remain for the major part an unanswered research question.

5 Classification of Existing Approaches

In this section we consider and classify according to the variants identified in the present chapter some of the existing approaches proposed in the literature, and aiming to prove regulatory compliance of business process models. We organise the approaches according to the main technique used to solve the problem of proving regulatory compliance.

5.1 Control-Flow Based

These approaches focus on checking the execution order of the tasks in the business process models. To do so, temporal logic is generally used to verify properties over the execution orders of the tasks, some instances of this type of solution have been provided by Awad et al. [4], and by Lu et al. [48]. As the properties expressed through temporal logic formulae apply through the whole extents of the executions of the model, and refer directly to the tasks[7], then we can assign these approaches to the variant **nG-** of the problem.

Other approaches based on the control-flow can use different formalisms to represent the constraints between the execution order of the tasks. One of such approaches proposed by Groefsema et al. [34], uses CTL^*[8] to describe the constraints between the execution order of the tasks. As this approach allows to express conditional constraint, it can then be classified as belonging to the variant **nL-**. Mind

[7]Having constraints referring directly to the tasks, in this case where temporal logic is used, it means that some constraint over the execution order is expressed between two particular tasks. When this is the case, the constraints refer to the labels of the tasks, which can be considered as propositional literal.

[8]CTL* is a combination of computational tree logic and linear temporal logic, which allows to combine path quantifiers and temporal operators.

that the approach proposed by [34] adopts also some practical optimisation, as it does reduce the space-state while preserving the information contained in the concurrent components of the model. This provides only a practical optimisation for the problem, while the theoretical computational complexity still holds for its worst case scenarios.

5.2 Temporal Logic Related

The approach proposed by Elgammal et al. [18] introduces a new language to represent the execution constraints between the tasks of a business process model. This language, named *Compliance Request Language*, is used by [18] to define patterns that must be followed by the executions of the model. While the language proposed allows to compactly express these patterns, the authors shows that *Linear Temporal Logic* can be used to represent the patterns. As the constraints over the executions of the business process model are conditional, we can assign this approach to the variant **nL-**.

5.3 Classical Logic Related

Considering now approaches [45, 23, 41, 35] adopting classical logic formalisms to represent the constraints over the execution of the process model, the introduction of logical formulae allows to represent more complex conditions and constraints. For instance the constraint over the order execution of a task can be conditional with respect to the execution of a set of tasks, or the combined effects of a set of tasks must be achieved before a given deadline. Given the expressivity that can be achieved by such approaches, then we classify as belonging to the variant **nL+**.

5.4 Modal Logic Related

Other approaches, such as the ones proposed by Sadiq et al. [59], and Governatori [24], adopt modal logic to represent the constraints over the allowed execution orders of the tasks of a business process model. While the inclusion of modalities over classical logic based approaches allows to improve the expressivity of the constraints, the more expressive constraints are still compatible with the variant **nL+** of the problem.

5.5 Practical Optimisations

Given the inherent computational complexity of the problem, several approaches have adopted techniques allowing to reduce the search state-space of the problem to

Problem Variant	Approach
nG-	Control-Flow Based [4]
	Control-Flow Based [48]
nL-	Control-Flow Based using CTL* [34]
	Temporal Logic [18]
	Practical Optimisation [53]
	Practical Optimisation [8]
	Practical Optimisation [60]
	Practical Optimisation [20]
	Practical Optimisation [39]
nL+	Classical Logic [45]
	Classical Logic [23]
	Classical Logic [41]
	Classical Logic [35]
	Modal Logic [59]
	Modal Logic [24]

Table 4: Classifying Existing Approaches

limit the state explosion in concurrent processes. However, these approaches either generate large amounts of overhead, such as for instance the one introduced by Nakajima [53], or lose information on concurrency and the orders of local tasks due to the linearisation of the concurrent components of the process model, as shown in the following approaches [8, 60, 20, 39]. We classify the approaches based on practical optimisations as belonging to the variant **nL-**.

5.6 Summary

We conclude this section by summarising the classification of some of the existing approaches in Table 4.

The first thing that can be noticed is that every single approach falls into the **NP**-complete computational complexity class when the goal is to prove partial compliance of a business process model. Differently, if the aim is to verify whether a model is fully compliant with the given regulations, then the logic based approaches are in co**NP**-complete, while the others can be theoretically solved in polynomial time.

The second and final observation over Table 4, concerns the distribution of the approaches over the various variants of the problem. We would like to point out that when global ordering constraints, as given in *control-flow* based approaches, then

the problem lies in the **nG-** variant. Introducing conditional constraints, usually through the adoption of *temporal logics* or derivates, moves the problems into the **nL-** variant. Finally, when full fledged *logical formalisms* are used to represent the constraints, then the problem reaches the most difficult variant discussed in this chapter: **nL+**.

6 Conformance and Normative Reasoning

In this section we discuss some disciplines related to the problem of proving regulatory compliance of business process models. In particular we discuss *conformance*, verifying whether a trace from a log is a proper execution of a given process model, and normative reasoning, the discipline tasked with reasoning and deal with normative concepts, such as obligations and violations.

In addition to discuss the relations of these disciplines with the problem discussed in this chapter, we also discuss their computational complexity and how it relates to the results presented in this chapter for the problem of proving regulatory compliance.

6.1 Conformance Checking

Conformance checking, as defined by van der Aalst [69], refers to the discipline of verifying whether the executions contained in a given event log are the proper executions of a given process model. To put it differently, using van der Aalst words: "The goal is to find commonalities and discrepancies between the modeled behaviour and the observed behaviour."

While conformance checking and compliance checking are orthogonal disciplines, mainly related as both deal with business process models, both prove useful in verifying the properties of models and their actual behaviour, and used together allows to ensure the compliance of the actual behaviours of business process models in real life scenarios. Considering a business process model used by an organisation, its compliance can be verified by using one of the many available techniques. In particular, if we focus on the case where *full compliance* is being proven for such a model, what is verified is that every proper execution of the model is compliant with the regulatory framework in place. However, while this is indeed a desirable property of a business process model, as van der Aalst mentions, its not always the case that the modelled behaviour of a business process model in an organisation, perfectly reflects the actual observed behaviour of how such organisation performs its business. Therefore, *conformance* becomes extremely important to verify whether the actual behaviour follows the modelled one, ensuring in this way that the organisation is

compliant with the regulations. Moreover, when discrepancies are detected, it is desirable to realign the modelled behaviour of the organisation with the observed one, using available techniques such as for instance *process mining* [70][9], in such a way the compliance of the realigned model can be rechecked and if the actual behaviour of the organisation keeps following the realigned model, then regulatory compliance is assured.

Token Replay

A technique to verify conformance of traces in a given event log with respect to a business process model is by using token replay [58]. This technique, as the name suggests, consists of trying to replay the traces over the model. One thing to notice, is that this technique is originally designed to verify the conformance of event logs with workflow models based on petri nets, such as the approach proposed by [1], as described in Chapter 4, Part 2, Section 2.2. Despite this difference, the technique can be adapted to deal with business process models as well, especially given their similarities as pointed out by [44].

The original technique, replaying traces over workflow models constructed using petri nets, is based on going through the list of tasks representing the trace being checked whether it conforms with the workflow model. The workflow is setup having a token in its starting *place*, and each task in the trace is then checked to verify whether the current state of the model allows its execution, in accordance to the state of the tokens[10]. After, tokens in the precondition set are consumed and recreated in the postcondition set of the task in the model. This is repeated for every task composing the trace, and at the end of the analysis, every discrepancy detected, like missing required tokens to execute a task in the order defined by the trace, as well as remaining tokens in the model's places, with the exception of the final place[11], is considered to determine how much deviation there is between the observed behaviour and the expected behaviour of a model. Naturally, when the trace is a proper trace of the model, then no discrepancies are detected.

[9] With the term process mining, we refer to techniques capable of distilling a business process model fitting a given event log of traces. Such techniques can be used to construct a process model from scratch, or to adapt existing process models to properly fit the actual observed behaviour. While this is another relevant discipline related to business process models, we do not delve in its details in this chapter as it is only marginally related to the problem of proving regulatory compliance.

[10] As a reminder, in a Petri Net a task, also referred to as a transition, can be executed if every place in its precondition set contains a token.

[11] A workflow model based on Petri Nets, is considered to be sound if it is executable without leaving tokens within its internal places after the execution is concluded.

Data Driven Conformance

Understanding whether an execution, composed by simple sequence of tasks, belongs to a process model is important. Considering only the execution sequence may be in fact not enough to properly measure conformance. The execution of business process models involves additional factors, such as the state of the execution, in other words the data corresponding to the execution. This is represented in the present chapter as the process' states and associated to the execution of the tasks of the process in their corresponding traces.

Efforts towards this taking into account data while measuring conformance has already been made, as for instance by De Leoni et al. [16], which adopt an approach using A^* to calculate the alignment[12] between the trace being evaluated and the process model, which allows to evaluate data and resources in addition to the execution order of the tasks in a trace.

While De Leoni et al. [16] claim their approach to be sub-linear in time with respect to the size of the log and model being evaluated, it must be considered that being A^* heuristic based, hence trying to optimise the search space being investigated by smartly reducing it, there is always the possibility that part of the search space containing the optimal solution (or a solution) for the problem to be discarded. In general, conformance verification procedures are solvable in time polynomial with respect to the size of the problem, as for instance the approach proposed by Sun and Su [64], based on solving syntactic characterisations of some subclasses of *DecSerFlow* constraints[13].

Conformance and Legal Requirements

In addition to data, sometimes it is necessary to verify whether the actual behaviour of an organisation (its logs) conforms with the legal requirements in place. While business process regulatory compliance represents a way to indirectly verify this through its pairwise use with conformance checking, sometimes a more direct approach is desirable in particular to determine if the actual execution of instances of the process do not violate legal requirements. In this case, we can speak of run-time compliance (if checked while a process instance is executed) or auditing (if it is a post-mortem analysis of the instances). Generally, run-time compliance with the legal requirements can be handled with the same techniques adopted for design-time compliance. However, there are a few differences: the first is about the data to be

[12] Alignment is a measure related to conformance, and it measure how close, aligned, is a given execution with the possible executions of a given process model.

[13] DecSerFlow is an extensible language, which stands for: *declarative service flow* language. It can be used to specify and monitor service flows, in addition to verify their conformance.

used for the annotations. At design time, we do not know the actual value for the data (and most approaches assume annotations expressed as propositional/boolean variables) and those values must be instantiated by the actual value occurring in the instances of the processes. After the data has been instantiated, the theoretical complexity of auditing is reduced to the complexity of the underlying logic/framework given that the number of states is linear and it is determined by the number of entries in the process log (and every instance corresponds to a single trace of the process model). For run-time compliance, the issue is whether one is interested to check if the current instance at the then current task is compliant, in which case the complexity is the same as the complexity of auditing (given that the problem is reduced to the case of a single trace); alternatively, one can check if it is possible to terminate the current instance with no violations or all possible terminations are compliant. Clearly, both cases reduce to the situation where we have to determine if a sub-process model is compliant; in particular the (sub-)process model obtained by the original process model where we delete all paths not passing from the current task, and identifying the start of the (sub-)process model with the current task. Hence, the problem of determining if there is a compliant termination is reduced to the case of partial compliance, and all possible termination are compliant to full compliance.

6.2 Normative Reasoning

Finally, after having discussed the relation between business process compliance and conformance, we discuss the further relations with the area of *Normative Reasoning*, tasked about reasoning about norms and normative concepts, and how they affect various type of environment. While many different formalisms/logics/frameworks have been proposed for normative reasoning, ranging from various deontic logics [22], different systems of non-monotonic reasoning [56, 40, 25], event calculus [61], Input/Output logic [50] and various forms of expert systems and AI and Law systems [5], the study of the complexity of legal and normative reasoning has been largely neglected. Despite this, the complexity classes for the different approach is well understood: modal logic [46, 36] for deontic logics, though, with almost no results for conditional and dyadic deontic logic[14]; complexity of default logic, argumentation [6] for non-monotonic reasoning, and ad hoc results for event calculus [7]. Practically, all approaches are **NP**-complete or with higher computational complexity. In what follows we briefly discuss some notable exceptions.

Some of the work dedicated to the complexity of normative reasoning concerns

[14]In general conditional logics received much less attention than their modal counterparts, for some complexity results see [3].

the investigation of the complexity of Input/Output logic [62, 63] where the complexity of some I/O variants is investigated also in connection to the representation of norms (including I/O with permissive norms; however, the work is dedicated to the study the complexity of logics, and not to normative reasoning problems. Most the problems (e.g., consistency, fulfilment) discussed by Sun and co-workers are not tractable. For example, consistency is some of the basic I/O logic (simple-minded I/O logic) is **NP**-complete, and fulfillment is **coNP**-complete, with higher complexity for constrained I/O logics.

The second area of research related to computational complexity and normative reasoning is the work on Defeasible Deontic Logic. Contrary to the work reported above the Defeasible Deontic Logic (also known as PCL[15] [31]), is computationally feasible. [29] and [26] extended the result by [49] proving that the extensions of Defeasible Logic with deontic operators and violation operator of [27] is still computationally feasible, and the extension of a defeasible theory can be computed in **P**, more specifically, the problem is linear in the size of the theory, where the size of a theory is given by the number of rules and instances of literals in the theory. The result was further extended to included permission and weak permissions [25]. Similarly, [33] prove that temporalising PCL, allowing for explicit deadlines, and compensation does not increase the complexity of the logic, and the temporal extension can still be computed in time linear to the size of the theory, where, in this case, the size depends also on the distinct instants of time explicitly appearing in the given theory; this extends the result in [32].

[26] applied the work to the execution of business contracts, thus the performance of a contract can be executed in linear time. Furthermore, they discussed the issue of comparing contracts and proposed a normalisation procedure to this end. However, they did not investigate the complexity of the normalisation problem. [33] address this issue in the context of a temporal extension of the logic, and while they do not give a complexity result they provide an (exponential) upper-bound. Accordingly, they conjecture the problem to be computationally hard but argue that it might not be a problem for real life applications since the problematic cases are typically not very frequent and limited in the number of parameters.

In [28] the logic was used for modelling agents, in particular to the modelling of the so called social agents, i.e., agents where there is a conflict between one of their intention and a norm, they give an higher preference to the norm, dropping thus the conflicting intention. However, Governatori and Rotolo shown that there are situations where, even for social agents, adhering to the agent plan ends up in a non-compliant situation. Accordingly, the restoring sociality problem is to identify

[15] Process Compliance Logic.

a set of agent's intentions to drop to prevent the agent's plan to be non-compliant. Governatori and Rotolo proved that when norms and agents are represented using Defeasible Deontic Logic the restoring sociality problem is **NP**-complete. The logic employed in [28] can be used to model business processes, after all, one can consider a business process as the set of traces, where each trace is a sequence of task, where the annotations corresponds to the effects of the tasks, and the states include the preconditions of the tasks. Hence, the plan library of an agent can be understood as business process (or a set of business processes), where the intentions and the facts of a theory determine what are the traces/processes/sub-processes to be executed. Accordingly, the restoring sociality problem can be seen as a special case to recovery from non-compliance for business processes (in the **nL-** space).

7 Summary and Open Problems

In this chapter we focused our attention on the computational complexity of proving regulatory compliance of business process models. We first describe the variants of the problem by reusing the same classification used by Colombo Tosatto et al. [11]. After discussing the existing computational complexity results, we moved to discussing neighbouring areas which still require much investigation, in order to understand the computational complexity of a broader spectrum of the variants of the problem.

Finally, we conclude this chapter by listing the open problems identified.

7.1 Proving Partial Compliance for the Variant 1L-

This particular variant of the problem of proving regulatory compliance, identified by Colombo Tosatto et al. [11], involves verifying whether a structured business process model is compliant with a single conditional obligation whose parameters are represented by using propositional literals.

While Colombo Tosatto et al. proposed a conjecture, reported in Conjecture 2, stating that the variant **1L-** should be able to be solved in time polynomial with respect to the size of the problem, we proposed in the present paper the opposite conjecture in Conjecture 1. However, no formal proof have been provided to show that the conjecture is correct. Therefore, proving that either **1L-** belongs to the computational complexity class **NP-c**, or to the class **P**, remains a problem to be solved.

Conjecture 2 (**1L-** is in **P**)**.** *We currently have no information about the computational complexity of **1L-**. That is, we cannot infer its belonging to a computational*

*complexity class in a similar way as for **nG+**, as in this case the simpler variant (**1G-**) is in **P**.*

*Our conjecture is that the computational complexity of **1L-** is in **P**. We have proven that moving from **-** to **+**, or from **1** to **n**, definitely brings the complexity into **NP-c**. In general, solutions tackling such variants have to explore the entire set of possible executions in the worst case scenarios, which precludes efficient solutions. Despite moving from **G** to **L** seems to definitely increase the complexity, we strongly believe that it does not influence the computational complexity of the problem enough to move it into **NP-c**, and polynomial solutions are still possible.*

7.2 Proving Regulatory Compliance of Unstructured Process Models

While the computational complexity of proving regulatory compliance of structured business process models has been extensively studied, the same cannot be said for unstructured process models. Therefore, a thorough analysis of the computational complexity for these unstructured variants of the problem is still an open problem. Considering that unstructured process models become structurally very similar to petri nets, investigating this similarity can be the initial step towards this analysis.

Moreover, as currently for structured process models, the variants of the problem are identified solely on adopting different properties of the regulatory framework being used to check regulatory compliance, identifying a set of structural properties of the models would allow to identify additional variants of the problem on the top of the ones already identified. The advantage in this case could be to allow a *divide and conquer* approach for studying the computational complexity of the problem, and possibly identifying simpler and harder versions of the problem.

Finally, given the relations with petri nets, correlating the structural properties of the variants of the compliance problems involving unstructured processes with known issues of petri nets (i.e., the undecidability of coloured Petri Nets) may be able to provide interesting results, which can potentially benefits both problems.

7.3 The Impact of Loops

Loops can be included in the business process models to improve the expressivity of the problem, allowing the repeated execution of tasks. Despite the obvious usefulness of including these type of constructs in process models, how much their inclusion increases the computational complexity of the problem in either structured and unstructured variants has not yet been studied.

As discussed in this chapter, introducing loops in business process models brings

them closer to *complete Turing machines*, which also leads to the inheritance of the problems affecting them (i.e., undecidability due to the *halting problem*). Therefore, as mentioned while discussing the open problems related to proving compliance of unstructured processes, identifying a set of properties allowing to identify a relevant number of variants can help in the computational complexity analysis, as well as allowing to identify, if it exists, the line between these problem's variants separating the ones which can be considered complete Turing machines from the ones which cannot.

7.4 Conformance

While only tangentially related to the problem of proving regulatory compliance of business process models, the solutions for these problem can be used in combination to ensure stronger properties. While the computational complexity of verifying conformance has been shown to not be a big obstacle for the problems considered, their scope can be definitely broadened to cover more interesting variants, such as for instance considering the regulatory requirements provided by a regulatory framework while conformance is being verified, where the additional challenges are related to the data (how to ensure that the "concrete" data at run-time/auditing correspond to the "abstract" data specified in the annotations of the tasks. In addition, normative requirements can span across multiple instances of the process (and multiple processes) and, in general, the instances are not synchronised.

References

[1] Arya Adriansyah, Boudewijn F. van Dongen, and Wil M.P. van der Aalst. Towards robust conformance checking. In *International Conference on Business Process Management*, pages 122–133. Springer, 2010.

[2] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50(2):510–530, 1985.

[3] Régis Alenda, Nicola Olivetti, and Gian Luca Pozzato. Nested sequent calculi for normal conditional logics. *Journal of Logic and Computation*, 26(1):7–50, 2016.

[4] Ahmed Awad, Gero Decker, and Mathias Weske. Efficient compliance checking using BPMN-Q and temporal logic. In *International Conference on Business Process Management*, pages 326–341. Springer, 2008.

[5] Trevor Bench-Capon, Michał Araszkiewicz, Kevin Ashley, Katie Atkinson, Floris Bex, Filipe Borges, Daniele Bourcier, Paul Bourgine, Jack G. Conrad, Enrico Francesconi, et al. A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. *Artificial Intelligence and Law*, 20(3):215–319, 2012.

[6] Marco Cadoli and Marco Schaerf. A survey of complexity results for non-monotonic logics. *The Journal of Logic Programming*, 17(2-4):127–160, 1993.

[7] Luca Chittaro and Alberto Montanari. Efficient temporal reasoning in the cached event calculus. *Computational Intelligence*, 12(3):359–382, 1996.

[8] Yongsun Choi and J. Leon Zhao. Decomposition-based verification of cyclic workflows. In *Automated Technology for Verification and Analysis*, pages 84–98. Springer, 2005.

[9] Silvano Colombo Tosatto. *Proving Regulatory Compliance: Business Processes, Logic, Complexity*. PhD thesis, University of Luxembourg and Università di Torino, 2015.

[10] Silvano Colombo Tosatto, Guido Governatori, and Pierre Kelsen. Business process regulatory compliance is hard. *IEEE Transactions on Services Computing*, 8(6):958–970, 2015.

[11] Silvano Colombo Tosatto, Guido Governatori, and Nick R.T.P. van Beest. Checking regulatory compliance: Will we live to see it? 09 2019.

[12] Silvano Colombo Tosatto, Guido Governatori, and Nick R.T.P. van Beest. Business process full compliance with respect to a set of conditional obligation in polynomial time. https://arxiv.org/abs/2001.10148, 1 2020.

[13] Silvano Colombo Tosatto, Guido Governatori, and Nick R.T.P. van Beest. Proving regulatory compliance: A comprehensive computational complexity analysis. *TBD*, 2021 forthcoming.

[14] Silvano Colombo Tosatto, Pierre Kelsen, Qin Ma, Marwane El Kharbili, Guido Governatori, and Leendert W.N. van der Torre. Algorithms for tractable compliance problems. *Frontiers of Computer Science*, 9(1):55–74, 2015.

[15] Byron Cook, Alexey Gotsman, Andreas Podelski, Andrey Rybalchenko, and Moshe Y. Vardi. Proving that programs eventually do something good. In *Proceedings of the 34th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '07, pages 265–276, New York, NY, USA, 2007. ACM.

[16] Massimiliano de Leoni, Wil M.P. van der Aalst, and Boudewijn F. van Dongen. Data- and resource-aware conformance checking of business processes. In Witold Abramowicz, Dalia Kriksciuniene, and Virgilijus Sakalauskas, editors, *Business Information Systems*, pages 48–59, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[17] Ivaylo Dobrikov, Michael Leuschel, and Daniel Plagge. Ltl. pages 204–211, 07.

[18] Amal Elgammal, Oktay Turetken, Willem-Jan van den Heuvel, and Mike Papazoglou. Formalizing and appling compliance patterns for business process compliance. *Software & Systems Modeling*, 15(1):119–146, 2016.

[19] Javier Esparza. On the decidability of model checking for several μ-calculi and petri nets. In Sophie Tison, editor, *Trees in Algebra and Programming — CAAP'94*, pages 115–129, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

[20] Sven Feja, Andreas Speck, and Elke PulvermÃijller. Business process verification. In *GI Jahrestagung*, pages 4037–4051, 2009.

[21] Nissim Francez. *Fairness*. Springer-Verlag, Berlin, Heidelberg, 1986.

[22] Dov Gabbay, Jeff Horty, Xavier Parent, Ron van der Meyden, and Leendert W.N.

van der Torre, editors. *Handbook of deontic logic and normative systems*. College Publication, 2013.

[23] Aditya Ghose and George Koliadis. Auditing business process compliance. In *ICSOC 2007*, pages 169–180, 2007.

[24] Guido Governatori. The Regorous approach to process compliance. In *2015 IEEE 19th International Enterprise Distributed Object Computing Workshop*, pages 33–40. IEEE, 2015.

[25] Guido Governatori, Francesco Olivieri, Antonino Rotolo, and Simone Scannapieco. Computing strong and weak permissions in defeasible logic. *Journal of Philosophical Logic*, 42(6):799–829, 2013.

[26] Guido Governatori and Duy Hoang Pham. DR-CONTRACT: an architecture for e-contracts in defeasible logic. *International Journal of Business Process Integration and Management*, 4(3):187–199, 2009.

[27] Guido Governatori and Antonino Rotolo. Logic of violations: A Gentzen system for reasoning with contrary-to-duty obligations. *Australasian Journal of Logic*, 4:193–215, 2006.

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

[29] Guido Governatori and Antonino Rotolo. A computational framework for institutional agency. *Artificial Intelligence and Law*, 16(1):25–52, 2008.

[30] Guido Governatori and Antonino Rotolo. A conceptually rich model of business process compliance. In Sebastian Link and Aditya Ghose, editors, *7th Asia-Pacific Conference on Conceptual Modelling*, volume 110 of *CRPIT*, pages 3–12. ACS, 2010.

[31] Guido Governatori and Antonino Rotolo. Norm compliance in business process modeling. In *Proceedings of the 4th International Web Rule Symposium: Research Based and Industry Focused (RuleML 2010)*, volume 6403 of *LNCS*, pages 194–209. Springer, 2010.

[32] Guido Governatori and Antonino Rotolo. Computing temporal defeasible logic. In *RuleML 2013*, pages 114–128, 2013.

[33] Guido Governatori and Antonino Rotolo. Time and compensation mechanisms in checking legal compliance. *Journal of Applied Logics – IFCoLog Journal of Logics and their Applications*, 6(5):817–847, 2019.

[34] Heerko Groefsema, Nick R.T.P van Beest, and Marco Aiello. A formal model for compliance verification of service compositions. *IEEE Transactions on Services Computing*, 11(3):466–479, 2018.

[35] Stephan Haarmann, Kimon Batoulis, and Mathias Weske. Compliance checking for decision-aware process models. In *International Conference on Business Process Management*, pages 494–506. Springer, 2018.

[36] Joseph Y. Halpern and Yoram Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial intelligence*, 54(3):319–379, 1992.

[37] Mustafa Hashmi, Guido Governatori, Ho-Pun Lam, and Moe Wynn. Are we done with business process compliance: State-of-the-art and challenges ahead. *Knowledge and Information Systems*, 01 2018.

[38] Mustafa Hashmi, Guido Governatori, and Moe Thandar Wynn. Normative requirements for regulatory compliance: An abstract formal framework. *Information Systems Frontiers*, 18(3):429–455, 2016.

[39] Jörg Hoffmann, Ingo Weber, and Guido Governatori. On compliance checking for clausal constraints in annotated process models. *Information Systems Frontiers*, 14(2):155–177, 2012.

[40] John F. Horty. Deontic logic as founded on nonmonotonic logic. *Annals of Mathematics and Artificial Intelligence*, 9(1-2):69–91, 1993.

[41] Conrad Indiono, Walid Fdhila, and Stefanie Rinderle-Ma. Evolution of instance-spanning constraints in process aware information systems. In *OTM Confederated International Conference "On the Move to Meaningful Internet Systems"*, pages 298–317. Springer, 2018.

[42] Gerhard Keller and Thomas Teufel. *SAP R/3 Process Oriented Implementation*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1998.

[43] Yonit Kesten, Amir Pnueli, Li-On Raviv, and Elad Shahar. Model checking with strong fairness. *Formal Methods in System Design*, 28(1):57–84, Jan 2006.

[44] Bartek Kiepuszewski, Arthur H.M. ter Hofstede, and Christoph Bussler. On structured workflow modelling. In *Proceedings of the 12th International Conference on Advanced Information Systems Engineering*, CAiSE '00, pages 431–445, London, UK, UK, 2000. Springer-Verlag.

[45] David Knuplesch, Linh Thao Ly, Stefanie Rinderle-Ma, Holger Pfeifer, and Peter Dadam. On enabling data-aware compliance checking of business process models. In *International Conference on Conceptual Modeling*, pages 332–346. Springer, 2010.

[46] Richard E. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM journal on computing*, 6(3):467–480, 1977.

[47] Niels Lohmann, Eric Verbeek, and Remco Dijkman. *Petri Net Transformations for Business Processes – A Survey*, pages 46–63. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[48] Ruopeng Lu, Shazia Sadiq, and Guido Governatori. Compliance aware business process design. In *International Conference on Business Process Management*, pages 120–131. Springer, 2007.

[49] Michael J. Maher. Propositional defeasible logic has linear complexity. *Theory and Practice of Logic Programming*, 1(6):691–711, 2001.

[50] David Makinson and Leendert W.N. van der Torre. Permission from an input/output perspective. *Journal of philosophical logic*, 32(4):391–416, 2003.

[51] Mediatrix Makungu, Michel Barbeau, and Richard St-Denis. Synthesis of controllers of processes modeled as colored petri nets. *Discrete Event Dynamic Systems*, 9:147–169, 05 1999.

[52] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[53] Shin Nakajima. Verification of Web service flows with model-checking techniques. In *Proceedings of First International Symposium on Cyber Worlds*, pages 378–385, 2002.

[54] Artem Polyvyanyy, Luciano García-Bañuelos, Dirk Fahland, and Mathias Weske. Maximal structuring of acyclic process models. *The Computer Journal*, 57, 01 2014.

[55] Artem Polyvyanyy, Luciano García-Ba nuelos, and Marlon Dumas. Structuring acyclic process models. *Information Systems*, 37(6):518 – 538, 2012.

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

[57] PWC. *2017 Risk and Compliance Benchmarking Survey*, 2017.

[58] Anne Rozinat and Wil M.P. Van der Aalst. Conformance checking of processes based on monitoring real behavior. *Information Systems*, 33(1):64–95, 2008.

[59] Shazia Sadiq, Guido Governatori, and Kioumars Namiri. Modeling control objectives for business process compliance. In *International conference on business process management*, pages 149–164. Springer, 2007.

[60] Shazia Sadiq, Maria E. Orlowska, and Wasim Sadiq. Specification and validation of process constraints for flexible workflows. *Information System*, 30(5):349–378, 2005.

[61] Marek J. Sergot, Fariba Sadri, Robert A. Kowalski, Frank Kriwaczek, Peter Hammond, and H. Terese Cory. The british nationality act as a logic program. *Communications of the ACM*, 29(5):370–386, 1986.

[62] Xin Sun and Diego Agustín Ambrossio. Computational complexity of input/output logic. In Antonis Bikakis and Xianghan Zheng, editors, *Multi-disciplinary Trends in Artificial Intelligence - 9th International Workshop, MIWAI 2015, Fuzhou, China, November 13-15, 2015, Proceedings*, volume 9426 of *Lecture Notes in Computer Science*, pages 72–79. Springer, 2015.

[63] Xin Sun and Livio Robaldo. On the complexity of input/output logic. *Journal of Applied Logic*, 25:69–88, 2017.

[64] Yutian Sun and Jianwen Su. Conformance for decserflow constraints. In Xavier Franch, Aditya K. Ghose, Grace A. Lewis, and Sami Bhiri, editors, *Service-Oriented Computing*, pages 139–153, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[65] A. M. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 01 1937.

[66] Wil M.P. van der Aalst. *A class of Petri nets for modeling and analyzing business processes*. Computing science reports. Technische Universiteit Eindhoven, 1995.

[67] Wil M.P. van der Aalst. Verification of workflow nets. In *Proceedings of the 18th International Conference on Application and Theory of Petri Nets*, ICATPN '97, pages 407–426, London, UK, 1997. Springer-Verlag.

[68] Wil M.P. van der Aalst. The application of petri nets to workflow management. *Journal of Circuits, Systems, and Computers*, 8(1):21–66, 1998.

[69] Wil M.P. van der Aalst. Distributed process discovery and conformance checking. In Juan de Lara and Andrea Zisman, editors, *Fundamental Approaches to Software Engineering*, pages 1–25, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[70] Wil M.P. van der Aalst. *Process Mining: Data Science in Action*. Springer, Heidelberg, 2 edition, 2016.

 Received 1 February 2021

An Exploratory Study on the Use of Artificial Intelligence to Initiate Legal Understanding for Business Development

Alessia Grassi
University of Huddersfield, Queensgate HD1 3DH
agrassi@hud.ac.uk

Mauro Vallati
University of Huddersfield, Queensgate HD1 3DH
m.vallati@hud.ac.uk

Abstract

Given the dynamic environment and the ever-changing international context, it is pivotal for companies to be able to quickly and effectively identify potential threats and opportunities. This can be done via environmental scanning, that allows to develop potential scenarios which help in proactively plan responses to potential risks. Yet, the process of scanning, and the design and analysis of scenarios, is extremely expensive, as it has to be done manually. Therefore, they cannot be exploited as often as they should to deliver the maximal benefit to a company.

In this chapter, we propose the use of Artificial Intelligence (AI) techniques to support the PESTLE analysis, a managerial tool used to identify those external factors which might affect a company. In particular, we focus on one of the environments scanned through PESTLE, the legal environment, and how AI can support this time and labour consuming process.

1 Introduction

Companies operate in an ever-changing and dynamic environment. It is not sufficient being able to react to these changes, in order to be competitive and capable to provide consumers with the right product or service. By the time a business has reacted on the information collected, and has implemented the right strategy to react to a change, it would be probably too late. The most effective way to maintain

a company competitive is to dynamically anticipate these changes. By undertaking environmental scanning, from which it is possible to develop scenarios, this might be successfully achieved [12]. There are evident limitations in scanning the environment, and one of this is the approach utilised which might vary from significantly organised and principled to totally random. This not only influences the kind of information which is possible to collect and exploit, but also the level of costs which the business has to tackle. To avoid waste of resources, being aware of what is needed is key to make the most of environmental scanning and scenarios strategies. This chapter is focusing on one specific approach widely utilised in business management to help articulate the macro-factors which can influence a business, the PESTLE analysis. This framework considers Political, Economical, Social, Technological, Legal, and Environmental factors as the main external macro-factors influencing a business strategy.

The recent advancement in technology, and especially the growing abilities of Artificial Intelligence (AI) techniques, might facilitate businesses in collating information and ease the scanning and strategising process. There has been, in particular, a significant amount of work in AI for dealing with laws and regulations. Dedicated conferences and workshops, such as JURIX[1] and ICAIL[2], and journals such as *Artificial Intelligence and Law* (Springer) focus on the design and development of AI approaches fit for the purpose of analysing and processing legal documents. Recent related work in the area focused, for instance, on the newly introduced EU General Data Protection Regulation, and include the use of machine learning systems to automatically check the compliance and adequacy of privacy policies [16], and the introduction of a dedicated GDPR ontology [26]. More general works include the possibility to query a knowledge base of laws, in order to be provided with relevant paragraphs as answer [15]; the automated analysis of the interaction between legal systems in terms of possible arising conflicts for agents that have to operate in accordance to them [21]; and the use of a compliance management framework to expressively represent the specifications of normative requirements that impose constraints on various activities of a business process [18]. On related topics, there has also been a significant amount of work in investigating the use of AI approaches in courts [5, 9].

Here the authors discuss the possible benefits deriving from the exploitation of AI techniques to one of the macro-factors considered in the PESTLE analysis, the legal environment. This specific environment was chosen because it presents clear opportunities of classifications. Thus, it might foster the design of a modular ap-

[1] `http://jurix.nl`

[2] `https://icail2019-cyberjustice.com`

proach, where each module is capable of addressing the needs of a specific class, while maintaining flexibility and extensibility. Each module can therefore be designed and developed in isolation, and the most appropriate approaches can be exploited according to needs and requirements. The main contribution, beside the classification of main aspects that businesses have to deal with in regards to the legal environment, is the analysis of potentially-suitable AI methods, and the description of an overall framework that can be used as a basis for the design of the aforementioned modules.

The remainder of this paper is organised as follows. The next section further explores the PESTLE framework and its importance in businesses' decision-making process. Section 3 gives details of the legal environment, and Section 4 introduces the proposed classification of legal aspects to be dealt with in the context of the PESTLE analysis. The modular AI framework is described in Section 5. Finally, Section 6 gives the conclusions and highlights future works in the area.

2 The PESTLE Analysis

Despite being called PESTLE analysis, PESTLE is not an analysis instrument but more a framework used to remember all the important factors which might influence an environment. In particular, it is one of the most exploited models used to analyse the main variables which might influence the company's decision-making processes at a macro level. [23]. In particular, the name of the model is an acronym for the six external variables –the first version included only four– which are fundamental considering when developing or expanding a business: Political, Economical, Social, Technological, Legal, and Environmental (Figure 1). Companies utilise this model to investigate macroeconomics changes which are uncontrollable and unavoidable [7]. By being able to identify, investigate, and classify the impact of all these variables, managers are facilitated in identifying eventual threats which are not possible to directly control, evaluate potential high risks, and anticipate and limit these risks by developing possible future scenarios [6]. By doing so companies can conceptualise different scenarios, and develop potential alternatives. The model is particularly utilised when business are developing new products, or are expanding in new countries or new markets. However, it is also used on a regular base for understanding markets dynamics and cycles, and as such to eventually evaluate the position, potential and direction for a business [8].

Currently, PESTLE analysis is still performed manually: human experts have to collect relevant information and analyse them in order to suggest the most promising strategy to adopt in order to achieve the company's goals. Political and Legal fac-

tors are particularly demanding to investigate in terms of human labour and costs. Different countries and industries have specific bodies of law and principles which regulate any aspects of their market, contracts, business and stakeholders relationship. Moreover, these bodies of regulations tend to be updated and revised often, and even keep control of the changes requires some effort. Big companies inclusive of internal legal teams might face an easier process when collecting and analysing information aimed at developing new business opportunities. Yet, the time-cost and labour-cost for the collection and analysis of information is still high. For what concerns small companies, to time-cost needs to be added the cost of relaying on external legal experts which might facilitate gathering and analysing the information needed. On top of these costs, when entering new countries other barriers might complicate the process for both big and small companies, such as the language. There are organisations (private and governmental) which help small medium enterprises when exports are concerned: such as the JETRO to enter Japan, or Austrade to exit Australia [2]. In any case, even lawyers cannot always be knowledgeable of all the details concerning domestic, international and foreign legal aspects [28]. As a consequence, by being able to initiate an understanding of a legal environment without the need for employing a significant amount of resources, companies could drastically reduce the initial costs of these processes and positively effect their performances.

The use of AI techniques to scan the legal environment for information and potential threats might help scanning even the smallest creases of laws and regulations and help firms avoiding the risk of missing some critical aspects which a human being even if well trained might miss. The implementation of AI techniques in this scenario has to be considered as an aid for firms' initial explorations regarding new opportunities for business,to assist people and not to substitute them. AI could facilitate the planning for the launch of a new product, the opening of a new retail store, or even to prepare in view of important legislation changes as recently the mandatory implementation of the EU General Data Protection Regulation (GDPR)[3]. Finally, AI techniques could also facilitate the comparison between different potential strategies and help decide which one might be more effective under specific circumstances.

3 The Legal Environment

The number of legislation and regulations affecting businesses has significantly increased in the past decades, mainly due to global markets, and the expansion of e-commerce. Three are the main purposes that these legislation are attempting to achieve: protecting markets and society from irresponsible behaviours; protecting

[3] `https://eugdpr.org`

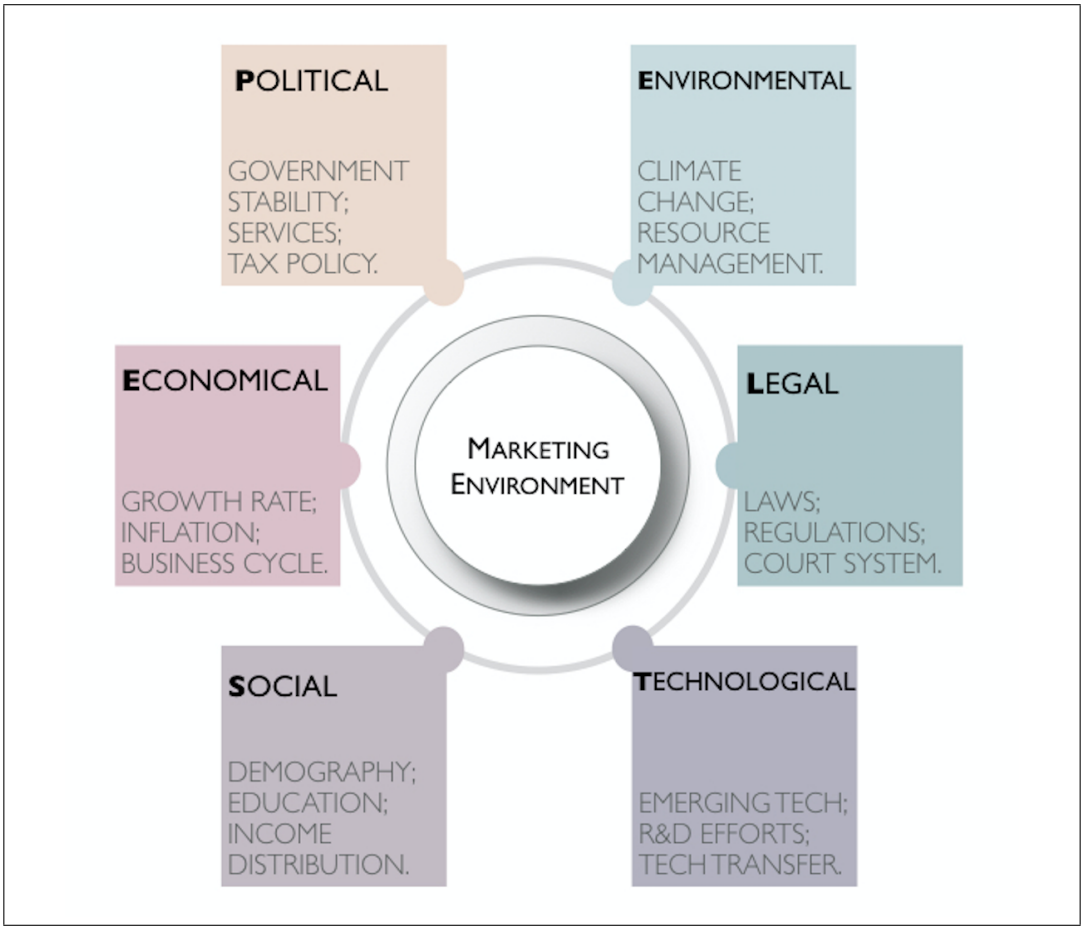


Figure 1: The PESTLE model, which emphasises the Political, Economical, Social, Technological, Legal, and Environmental aspects to take into account.

consumers' rights; and protecting companies from unfair competition [29]. It is important for companies and managers to be aware not only of current legislation and practices, but also of possible future development of public interest groups and legislation directions [29]. The possibility to utilise AI techniques to investigate the legal environment is an important perspective to avoid excessive costs at the beginning of a new process. This does not mean that firms will not need legal teams and consultants anymore. However, an initial scanning of the legislation might ease companies' processes, despite the fact that law bodies are still largely depending on human interpretation. It might be worth considering also that different cultures

have different way of approaching potential legal issues. For example in Japan although an increase in number of lawyers per capita, the demand for legal services is not as high as in other countries.[4]

As it will be properly analysed in section 4, one of the most difficult things to grasp in the legal environment is the different institutions and interest groups involved in a market and the hierarchical level they are operating at. There are national, international, and global legislation to considered, together with extra-jurisdictional bodies and institutions. Thus, it is pivotal to know who is jurisdictionally responsible at a specific level. There might be a national regulation in place regarding contracts, which might be override by an extra-national body. An example of international institution is UNCITRAL, a commission established by the United Nations to help in filling a communication gap between different countries and they created a Convention regarding contracts which is quite similar to an article of the United States Commercial Code, so to facilitate tradings between the US and other countries [28]. It is because of this diversification of bodies involved that this paper is providing an initial framework which collate the most significant levels to consider in a legal scanning.

3.1 Why AI might help the Legal Scanning

There are several reasons which brought the Authors to focus on the opportunities of applying AI techniques to a business tool such as PESTLE. On of these is that the number of news regarding law infringements committed by big corporations is exponentially increasing, especially due to the perfection of regulations concerning global markets and grey areas. In the UK there has been an increase of corporate fines up to 18 percent from 2016 to 2017 and only referring to health and safety regulations breach.[5] There are several other examples of law infringement: from trade mark, to tax evasion; from unfair competition, to violation of labours' and humans' rights. Not to mention law suites between corporations due to suspect copyright infringements and copycats. Apple vs Microsoft, Google vs Apple, Gucci vs Guess.[6] Of course many examples might lead back to unfair business behaviour, where corporations are well aware of their acting borderlines. Yet, there are many cases where law violations are due mainly to lack of knowledge and information. It is for these cases that the use of AI techniques might be of an aid and help to avoiding not only the waste of resources before starting the process, but also the

[4] https://blogs.wsj.com/briefly/2016/04/03/the-legal-industry-in-japan-the-numbers/

[5] https://www.hsmsearch.com/Corporate-fines-up-in-one-year

[6] https://realbusiness.co.uk/6-famous-copyright-cases

waste of money after due to having broken the law. It is important to consider that law violations have a significant financial repercussion on the business. However, another crucial aspect which derives from braking the law is the business' image damage. The bad publicity and PR which might derive from these episodes is sometimes even more dangerous than a fine and might permanently damage the ability of a firm to compete and generate profits.

The implementation of AI techniques in analysing the legal environment might prevent significant damage for companies. One of the most significant and recent example where AI might have helped in preventing significant losses is the breach of data protection. Many times these breaches happened because of the inability of a companies to adhere to the GDPR regulations: among others, Yahoo and Ebay were asked to pay billions for having breached GDPR requirements. Although it is one of the most nasty issue for businesses at the moment, data breach is only one of a numerous legal problems which a firm might need to face during its lifetime, and this is mainly due to the fast changing legal environment their are operating within. For example, Google was accused of tax evasion which derived from an uncertainty of the international tax system which is changing[7]; the Royal mail faced accusations of abuse of dominant position –breaking competition laws [8]; and Hugo Boss was caught in an health and safety breach[9]. In all the examples, the existence of an AI system would have greatly benefit these companies and would have allowed them to avoid paying fines and, as previously said, image damages.

4 A Legislative Classification Framework

As mentioned in the previous section, the legal environment is very complex and investigating it requires a significant effort of time and resources. Although there are numerous shades in any country legal system, it is possible to create an initial classification framework which describes the main layers a business needs to consider when approaching a new market (Figure 2). Far from being a mere academic exercise, this classification plays a major role in fostering the use of AI for the legislative environment in the context of the PESTLE analysis. As better detailed in Section 5, the notion of classes can be helpful in selecting the most relevant body of laws to be analysed and considered for the reasoning, and can also characterise the sort of

[7] https://www.theguardian.com/technology/2016/jan/22/google-agrees-to-pay-hmrc-130m-in-back-taxes

[8] https://www.theguardian.com/business/2018/aug/14/royal-mail-fined-competition-law-ofcom-whistl

[9] https://www.shponline.co.uk/news/breaking-hugo-boss-fined-1-2m-for-health-and-safety-breaches/

interaction that human experts may require to investigate corresponding aspects.

The idea behind the framework is to be a checklist or a road map for businesses through their initial legal scanning. Businesses are subject to numerous regulations, and all these regulations affect the businesses at different level. There are global agreements such as those implemented by the World Trade Organisation (WTO), but also extra-jurisdictional regulations such as those oversaw by the European Union, and finally national and local legislation issued by the hosting country. Furthermore, in some cases, companies are free to negotiate special agreements which might set them in a position of control over prices and competitors, allowing them to create entry barriers for other industries. These kinds of negotiations (and contracts) have to be considered as bodies of regulations which is not often easy to identify, especially if the company is new in the specific market. To facilitate the understanding of the different layers involved in a legal scanning a hierarchical framework was created as represented in Figure 2.

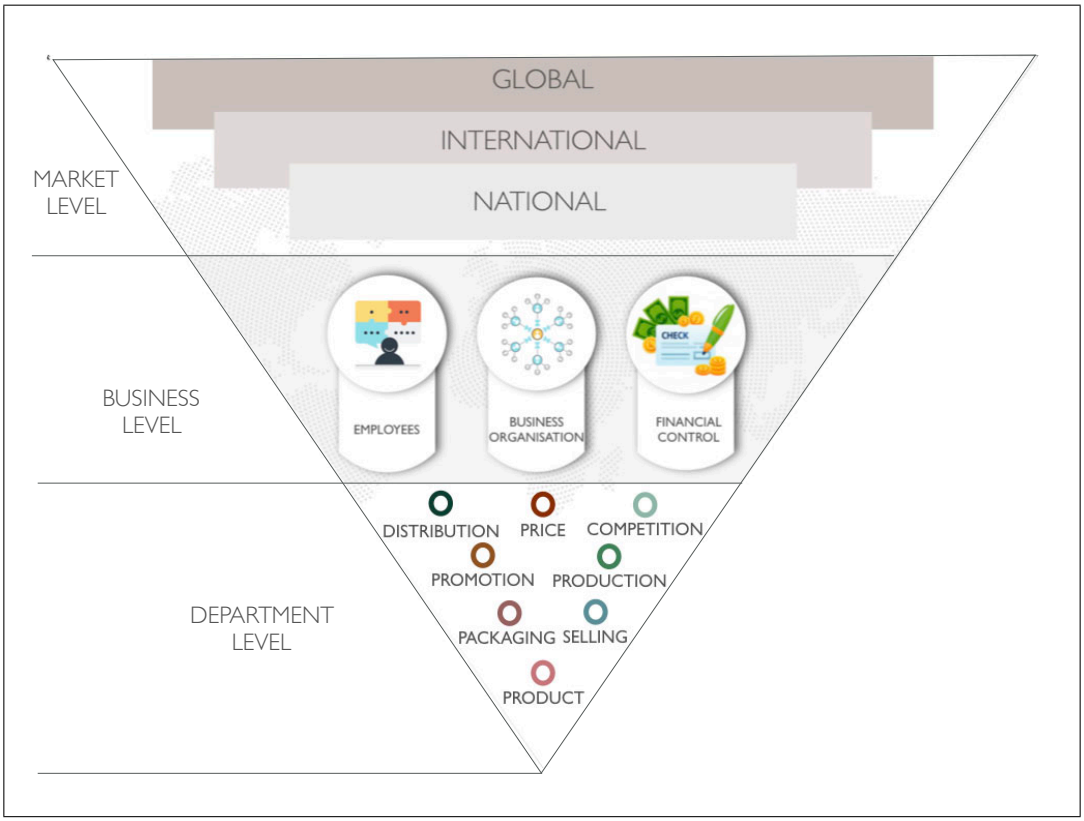


Figure 2: The classification of the legal environment into a hierarchical structure.

4.1 Market Level

4.1.1 Global Jurisdiction

The first issue a company might face when starting a legal scanning is to understand the hierarchical jurisdiction which is operating at market level. This is particularly difficult if the company is considering to operate in an unfamiliar market. For example, consumer protection and fraud prevention regulations are generally shifting the burden of responsibility between countries and international bodies. Hence, be aware of the boundaries and limitations of regulations is a first fundamental step to avoid nasty issues and possible law suits. By being able to understand the rules which apply in a specific market and their jurisdictional hierarchies, a company might save significant resources. Yet, quick reactions and adaptations to the complexity and the continuous changes of the legal landscape could be an extreme burden for some companies. On this matter, a straightforward exploitation of AI may automatically analyse the changes in the legal landscape, and assess whether the current policies used by the company are complying or not.

As represented in Figure 2, a general classification of this first level divides the market jurisdiction into global, international, and national. An example of how a regulation could become global is the application of the GSM standard when the mobile communication revolution started. In 1987 the EU circulated a technical mandatory standard to regulate hardware and services related to mobile communication technological transformation. The transformation quickly became global and the regulation generated large economies of scales effecting the entire world. Furthermore, as mentioned in section 3, in an effort of creating a globalised market, in 1966 the United Nation established the Commission on International Trade Law (UNCITRAL). In 1983, UNCITRAL developed the Convention on Contracts for the International Sales of Goods. This convention was an attempt of creating a single language among nations and harmonising different legal systems when regarding selling goods. Sometimes, as for the GSM example, not being able to quickly respond to changes in standards and regulations might turn into losses of market shares or of competitive advantages. Hence, it is fundamental to have a clear picture of what are the main forces operating on the market the business is attempting to approach.

4.1.2 International Jurisdiction

The same principle applies to international regulations, the second hierarchical layer at market level (Fig. 2). There are agreements and laws which can affect businesses which are operating in the European Union but, for example, not those which are

operating in China or in the US. One of the most significant, and recent, examples is the modernisation the European Union (EU) gave to the data protection regulations by revoking the Directive (95/46/EC) in 2016 through the introduction of what is widely known as the GDPR (General Data Protection Regulation - regulation 2016/679). This is a regulation which affects all businesses operating in the European Union boundaries, and requires businesses to have appropriate mechanism in place to guarantee consumers with the opportunity to decide on how and if businesses can use their personal data. However, US businesses operating on the US soil are not obliged to respect this regulation, unless they decide to start operating on EU markets. Since the introduction of this strict regulations, there has been numerous cases of businesses fined because in breach of the regulation, as mentioned in section 3. This is one of the clearer case in which the ability of quickly identifying loopholes in the business procedures might have spared business from monetary losses and image damages, where these kinds of breaches can also influence the kind of image a firm is projecting towards consumers by not respecting their privacy. Another fundamental aspect which often is regulated internationally to avoid anti-competitive practices is price. An example is the legal battle which Microsoft have fight against the EU with regard to specific components integrated in the operating system which the EU wanted the business to make available separately. This dispute went on for several years and cost the business precious resources in terms of time and especially of money. These kinds of regulations are fundamental in the attempt of maintaining a fair competitive environment, but yet, not being able to react quickly might cost businesses a fortune.

4.1.3 National Jurisdiction

The third and final hierarchical layer at market level is the national jurisdiction. There are numerous business aspects which are not regulated globally nor internationally; sometimes international bodies provide suggestions to the single countries, but then depend on the country the decision to implement or not the provided suggestion - as often happens in the EU. Examples of these kinds of legislations, to mention some, are: the opening hours of retail stores; the imposition of specific products' taxation such as VAT in UK or IVA in Italy; safety regulation of products and work environments; and patent and copyright.

4.2 Business Level

As previously discussed, there are numerous aspects to consider when initiating a legal scanning, and the jurisdiction is the first important layer to investigate. Once

the firm knows which bodies are regulating that aspect of the market, it is possible to go further in depth to a more specific level: the business level. As represented in Figure 2, this level can be divided in three macro-categories which encompass all major regulations which might impact the most significant business practices: Employees, Business Organisation, and Financial Control.

The Employees category regards all those laws and agreements which concern a business workforce. Part of this category are employees contract legislations which involve salaries, pension schemes, employees' benefits; but also working hours, annual leave regulations, work safety and environment conditions and so on. The second category is the business organisation category. Here are considered all those regulations which might influence the organisation structure of the business: from the business hierarchical organisation, to consumers' data management. The third and final category at business level is the financial control category. Part of this group are all those regulations regarding the financial aspect of a business: from tariffs, quotas and taxes to pay; to the way profits are managed and divided among shareholders.

4.3 Department Level

Finally, the department level represented in Figure 2 is the most specific of the three levels and exemplifies all the divisions which might be involved in the business. These regulations vary from specific legislations regarding advertisement content, to product safety, packaging and labels; from competitive behaviours, to price regulations; from trade marks and patents regulations, to selling techniques legislations.

To conclude this section, there are numerous aspects which need to be considered according to the market and the country the business is operating in. The framework provided in this chapter is an attempt of hierarchically categorise these aspects so to make it easier to design AI modules that can deal with them. Some regulations, as previously mentioned, might be established by a international organisation, some others might depend on the specific legislation of a single country, or might be a global agreement. Some agreements might affect the financial control of a business, or the human resources management; or might be even more specific and dictate specific opening hours for the retail store. By considering these three levels and all their specifications, it is safety to say that a business would be able to scan the legal environment and cover all the critical aspects involved.

5 A Modular Framework for Fostering the use of AI in Support of Legal Investigation for Businesses

The AI discipline has been increasingly turning its attention to the automated processing of complex information encoded in a non formal structure, as it is the case of laws and regulations. In fact, two main issues arise when dealing with such type of documents and knowledge: a large body of rules and regulations are not electronically stored; and these rules strongly rely on (potentially very different) interpretations, that can depend on the context or on other involved aspects. The first issue presents mostly barriers related to limiting the human effort which is necessary to encode and classify the knowledge stored in paper-based documents. However, there is a growing interest in encoding laws and regulations in some digital form, to ease the search and analysis (also for human experts). The latter issue is extremely challenging from an AI perspective, and is now object of significant research. On the one hand, it is envisaged that the use of machine learning –that can learn from past decisions and interpretations– may help on this matter. However, on the other hand, such techniques tend to lack of explainability, and can therefore lead to hard-to-understand and non-trustworthy results.

The large amount of work done in the area, particularly in the areas of Argumentation and Knowledge Representation, suggest that a large part of the process of the PESTLE analysis can be supported by AI-based agents. This is mainly the case of the process of capturing knowledge and reasoning on top of it, for supporting the subsequent human decision process. Leveraging on recent advancements, and on the classification introduced in Section 4, we are now in the position to introduce a framework for supporting legal investigation for businesses in the context of the PESTLE analysis.

The proposed framework depicted in Figure 3 includes the following four elements.

- **Legal documents**. There is of course the need of an appropriate corpus of legal documents, that can be encoded into suitable knowledge representation structures. On this matter, in order to reduce the burden on human experts, a number of automatic or semi-automatic approaches have been introduced. Examples include [10, 24]. Notably, it is not necessary to incorporate all the possible legal documents, but the classification provided in Section 4 has to be used to identify relevant articles. This approach can help in reducing the size of the data to analyse, and foster the use of more complex and more demanding AI techniques in the other modules.

- **Ontology**. Ontologies represent the standard way to model the knowledge re-

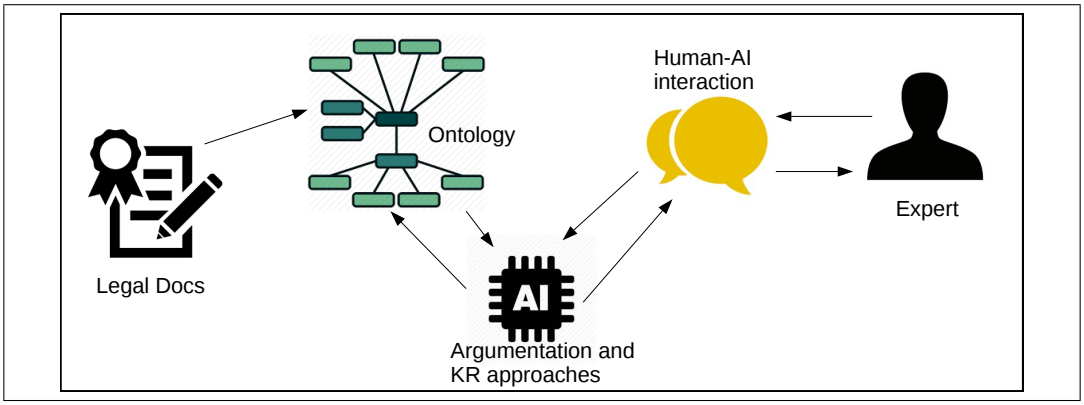


Figure 3: The proposed framework for fostering the use of AI in legal reasoning for businesses in the PESTLE context.

garding specific domains. In a nutshell, an ontology provides a structured way to store, process, and search knowledge [27]. In a typical ontology, *entities* can be defined, and relations between entities can be described and established. Furthermore, characteristics and attributes of each entity can be specified, so that an overall structure can be designed and exploited for processing purposes. An ontology has to be structured according to the specific kind of legal reasoning and knowledge that it has to support: it is not possible to design a single general ontology to be used for supporting any kind of reasoning. Thus, the classification provided in Section 4 has to guide the design of appropriate ontologies, according to the desired type of reasoning. In literature, several ontologies have been put forward to model specific kind of legal knowledge, and different methods have been introduced to compare this knowledge [20]. Examples of ontologies for the legal domain include [25, 26, 3, 13].

- **Query and Reasoning**. This module focuses on dealing with the requests made by users by querying the available ontology and performing appropriate reasoning. The knowledge stored in the ontology can then be analysed using argumentation approaches [11], so to provide an overview of the specific legal matter to the user, through paragraphs pointers and a first argumentative feedback. The field of argumentation provide means that can support automated reasoning. They propose arguments and counter-arguments to support or defeat a given statement, similarly to the way in which human experts would argue and debate. In that, conclusions reached by the approach can be easily investigated and explained, and the strength and validity of raised

arguments can be assessed. It should be noted that other approaches, for instance techniques based on ASP [4] or machine learning [15, 17] have also been investigated for querying purposes. The former class, similarly to argumentation, can provide answers to specific questions –potentially with explanations for the answer given. The latter is more focused on identifying paragraphs or articles where elements of potential interest for the user can be found. Here the reasoning is left to the human experts, but there is the advantage that the person can work directly on unstructured text.

- **Human-Computer Interaction**. The proposed framework involves and relies on human experts. It is not supposed to be fully autonomous, but to support the decision-making process of businesses. This architecture is designed to maintain humans in the loop, and to make sure that provided results are properly evaluated and assessed, in order to minimise risks for businesses. Given this perspective, it is pivotal to design appropriate means for supporting the interaction between humans and the system, and to support the explainability of answers provided by the system. This is in line with recent trends of research in the area [19, 1].

It is worth noting that the depicted AI framework has to be used as a blueprint to deal with different classes of the legislative framework identified in Section 4. This structure includes the main modules that have to be taken into account when dealing with any of the legislative classes. Yet, the actual techniques and approaches to exploit in each component can significantly vary according to the kind of questions the expert can pose, and the type of answers that are expected. There is a spectrum of potential interactions which might occur while implementing this framework, and the following two questions exemplify the two extremes:

- *What is the actual taxation in country X with regards to the class of products Y?* which requires a number (and possibly the corresponding legal article) as an answer; *Are our current internal regulations complying with the recently modified employment laws?* which requires a more complex answer, in terms of identified issues and corresponding laws and articles.

Intuitively, different classes of the legal framework may require only one type of interaction, with regards to the examples described above. In that case, the corresponding module could be tuned to the specific needs. In other cases, different types of interactions may be required, thus increasing the complexity of the AI modules. For instances, classes like price and promotion can be better suit for the first kind of interaction, while elements of financial control and employees rights may need the second type of interaction.

The human expert is still an important part of the process, and this is for two main reasons. First, the human expert can make sure that the analysis performed by the framework, as well as the provided reasons and arguments, is sound. It may of course be the case that some notions have been misinterpreted by the framework, or that some conclusions are based on, for instance, debatable or controversial articles and bodies of text. It is of pivotal importance that results obtained from the framework are reviewed by experts, and not blindly followed [14]. Second, the human should be able to interact with the framework in order to explore different scenarios and possibilities, either by changing the posed query or by objecting on some steps of the argumentative process.

6 Discussion and Conclusions

Companies are operating an extremely dynamic environment, where frequent and consistent changes to rules and regulations can have a disruptive impact on businesses if not timely and effectively dealt with. Companies can no longer just react to changes, but they are forced to constantly scan the environment to proactively identify threats and opportunities.

In this work, we investigated how AI can be used to support PESTLE analysis, which is a managerial tool widely utilised to articulate all those macro-factors which might influence a business. In particular, we focused on the legal environment, one of the most challenging due to large bodies of laws and regulations which have to be taken into account, and to its frequent changes. We introduced a classification of the aspects that companies have to deal with regards to legislation, and we positioned the different classes in terms of market, business, and department levels. The classification poses the pillars of the AI modular framework we introduced in Section 5. The framework incorporates all the relevant aspects that need to be taken into account in order to support companies when dealing with the legislation environment. Moreover, the classification helps in understanding the kind of interaction which is likely needed.

The proposed framework could not be extensively empirically tested, due to the required amount of information and strong involvement of companies and marketing experts. However, we had some qualitative discussions with PESTLE and marketing experts. Such discussions clearly indicate that an AI-based support for dealing with legal aspects would be favourably received by companies. As said, companies are struggling to quickly react to changes, and they are increasingly forced to take complex decisions based on limited knowledge.

Noteworthy, while it is clear that businesses can benefit from the exploitation of

AI, some scholars suggest that AI might also help consumers to protect themselves [22].

We see several avenues for future work. We plan to engage with experts who generally support companies in accessing new and international markets, such as the UK Department for International Trade[10]; or other similar departments and institutions. Such collaboration may lead to real-world case studies where the introduced frameworks can be validated. We are also interested in running some experiments with potential users, to better understand their requirements and needs. Finally, we are of course interested in extending the proposed framework to deal with all the other macro-factors considered by the PESTLE analysis tool, so to provide a comprehensive support system to perform environmental scanning.

Acknowledgements

This work has been partially supported by EU H2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 690974 for the project "MIREL: MIning and REasoning with Legal texts". The authors would also like to express their gratitude to Prof. Ken Satoh, National Institute of Informatics, Tokyo, Japan, for the useful discussions and support.

References

[1] Ashraf Abdul, Jo Vermeulen, Danding Wang, Brian Y. Lim, and Mohan Kankanhalli. Trends and trajectories for explainable, accountable and intelligible systems: An hci research agenda. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 582:1–582:18, 2018.

[2] Gerald S Albaum, Gerald Albaum, and Edwin Duerr. *International marketing and export management*. Pearson Education, 2008.

[3] BOER Alexander. Lkif core: Principled ontology development for the legal domain. *Law, ontologies and the semantic web: channelling the legal information flood*, 188:21, 2009.

[4] Theofanis Aravanis, Konstantinos Demiris, and Pavlos Peppas. Legal reasoning in answer set programming. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 302–306, 2018.

[5] Ryuta Arisaka, Ken Satoh, and Leendert W. N. van der Torre. Anything you say may be used against you in a court of law - abstract agent argumentation (triple-a). In *AI Approaches to the Complexity of Legal Systems - AICOL International Workshops 2015-2017: Revised Selected Papers*, pages 427–442, 2017.

[10] `https://www.gov.uk/government/organisations/department-for-international-trade`

[6] P. Baines, C. Fill, and S. Rosengren. *Marketing.* Oxford University Press, 2017.

[7] P. Baines, C. Fill, S. Rosengren, and P. Antonetti. *Fundamentals of marketing.* Oxford University Press, 2017.

[8] M. J. Baker. *Marketing strategy and management.* Palgrave Macmillan, 2014.

[9] Trevor Bench-Capon, Henry Prakken, and Giovanni Sartor. *Argumentation in Legal Reasoning*, pages 363–382. Springer US, 2009.

[10] Guido Boella, Luigi Di Caro, and Valentina Leone. Semi-automatic knowledge population in a legal document management system. *Artificial Intelligence and Law*, 27(2):227–251, 2019.

[11] Giorgio Bongiovanni, Gerald Postema, Antonino Rotolo, Giovanni Sartor, Chiara Valentini, and Douglas Walton. *Handbook of legal reasoning and argumentation.* Springer, 2018.

[12] Frances Brassington and Stephen Pettitt. *Principles of marketing.* FT Prentice Hall London, NY, 2005.

[13] Núria Casellas. *Legal ontology engineering: Methodologies, modelling trends, and the ontology of professional judicial knowledge*, volume 3. Springer Science & Business Media, 2011.

[14] Federico Cerutti, Alessia Grassi, and Mauro Vallati. Unveiling the oracle: Artificial intelligence for the 21st century. *Intelligent Decision Technologies*, 12(3):371–379, 2018.

[15] Diego Collarana, Timm Heuss, Jens Lehmann, Ioanna Lytra, Gaurav Maheshwari, Rostislav Nedelchev, Thorsten Schmidt, and Priyansh Trivedi. A question answering system on regulatory documents. In *Legal Knowledge and Information Systems - JURIX 2018: The Thirty-first Annual Conference*, pages 41–50, 2018.

[16] Giuseppe Contissa, Koen Docter, Francesca Lagioia, Marco Lippi, Hans-Wolfgang Micklitz, Przemyslaw Palka, Giovanni Sartor, and Paolo Torroni. Automated processing of privacy policies under the EU general data protection regulation. In *Legal Knowledge and Information Systems - JURIX 2018: The Thirty-first Annual Conference*, pages 51–60, 2018.

[17] Phong-Khac Do, Huy-Tien Nguyen, Chien-Xuan Tran, Minh-Tien Nguyen, and Minh-Le Nguyen. Legal question answering using ranking svm and deep convolutional neural network. *arXiv preprint arXiv:1703.05320*, 2017.

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

[19] Alexandra Kirsch. Explain to whom? Putting the User in the Center of Explainable AI. In *Proceedings of the First International Workshop on Comprehensibility and Explanation in AI and ML 2017 co-located with 16th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2017)*, 2017.

[20] Valentina Leone, Luigi Di Caro, and Serena Villata. Taking stock of legal ontologies: a feature-based comparative analysis. *Artificial Intelligence and Law*, 2019.

[21] Tingting Li, Tina Balke, Marina De Vos, Julian Padget, and Ken Satoh. Legal conflict

detection in interacting legal systems. In *Legal Knowledge and Information Systems - JURIX 2013: The Twenty-Sixth Annual Conference*, pages 107–116, 2013.

[22] Marco Lippi, Giuseppe Contissa, Francesca Lagioia, Hans-Wolfgang Micklitz, Przemysław Pałka, Giovanni Sartor, and Paolo Torroni. Consumer protection requires artificial intelligence. *Nature Machine Intelligence*, 1(4):168, 2019.

[23] del Thomas Marmol and Brigitte Feys. *PESTLE Analysis: Understand and Plan for Your Business Environment*. Namur: Lemaitre Publishing., 2015.

[24] Adeline Nazarenko, Francois Levy, and Adam Wyner. Towards a methodology for formalizing legal texts in legalruleml. In *Legal Knowledge and Information Systems - JURIX 2016*, volume 294, pages 149–154, 2016.

[25] Ha-Thanh Nguyen, Viet-Ha Nguyen, and Viet-Anh Vu. A knowledge representation for vietnamese legal document system. In *2017 9th International Conference on Knowledge and Systems Engineering (KSE)*, pages 30–35. IEEE, 2017.

[26] Monica Palmirani, Michele Martoni, Arianna Rossi, Cesare Bartolini, and Livio Robaldo. Legal ontology for modelling gdpr concepts and norms. In *Legal Knowledge and Information Systems: JURIX 2018: The Thirty-first Annual Conference*, volume 313, page 91, 2018.

[27] Steffen Staab and Rudi Studer. *Handbook on ontologies*. Springer Science & Business Media, 2010.

[28] Vern Terpstra, James Foley, and Ravi Sarathy. *International marketing*. Naper Press, 2012.

[29] Richard MS Wilson and Colin Gilligan. *Strategic marketing management*. Routledge, 2012.

 Received 1 February 2021