Legislative Knowledge Management with Property Graphs

Andrea Colombo[1,*], Francesco Cambria[1] and Francesco Invernici[1]

[1]Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Via Giuseppe Ponzio, 34, 20133, Milan, Italy

Abstract

The sparse nature and intricate set of relationships between legislative acts pose a significant challenge in the choice of the underlying database model, which both allows for performing structured queries and developing intuitive and smooth knowledge management. In this paper, we propose to use Property Graphs as a powerful alternative for managing legislative knowledge. First, we discuss how graph queries are a valid alternative solution to standard legislative knowledge management by showing how our data model fully captures the problem of law versioning (i.e., the existence of many versions for the same law). Then, we analyze, propose and implement innovative ways for monitoring the legislative system using Property Graph tools that have been recently standardized and developed, such as triggers and graph-based association rules, which empower our model of advanced ways of handling legislative data. For instance, we will show how we can use these tools to develop intelligent warning systems that inform stakeholders of critical changes in legislation through active rule reasoning or to detect shifts in graph patterns via continuous monitoring of significant association patterns.

Keywords

data management, property graphs, triggers, law

1. Introduction

Legislative data comprises the set of acts, bills, or other normative documents produced by an authority that detain a legislative power to regulate some domains on a national, regional, or international level. In most democratic countries, such a power is in the hands of parliaments, be they national, regional, or federal, according to the political system. One of the main challenges when dealing with such textual documents is converting them into machine-readable formats that enable the adoption of structured approaches when querying and analysing legislative acts.

To tackle this, the computer law community has devoted many efforts to proposing appropriate international standards to capture the common grounds of laws enacted in different legislative systems. Most of previous works have been based on the eXtensible Markup Language (XML) format, a semi-structured data model that has been naturally used for representing textual and hierarchical data, such as laws [1]. Relevant XML-based proposals include the Legal Knowledge Interchange Format (LKIF) [2], LegalRuleML [3] and Akoma Ntoso [4].

However, the need for a user-friendly storing of textual documents (with XML tags and hierarchical structures being ideal for such aims) collides with the possibility of conducting quantitative analysis of the complexities and features of the legislative system [5], which strongly depends on the ability to traverse relationships (e.g., dependencies and references between laws and articles). In fact, while the creation of XML formats to represent the components of an individual law (i.e., article or section structure, tags for references to other laws) is extremely useful for structured access to a document, it becomes cumbersome to manage and analyze trends and the complexity of the legislation. Furthermore, while XML-native databases generally support tools such as triggers [6], their focus on the document-level logic (e.g., handling events as document create, document update, document delete, property changes) harms the potential of *reasoning* over legislative data, leveraging relationships among acts to discover and detect anomalous patterns that require attention. For instance, abrogations of some portions of legislation can inadvertently create voids in legal foundations of third acts. Thus, a data model that cares more about relationships among acts is crucial in monitoring such scenarios.

In this paper, we present and discuss an innovative way of conducting legislative knowledge management based on Property Graphs (PG). By leveraging state-of-the-art advancements from the database community in the formal standardization of Property Graphs and their tools [7, 8, 9], we present an alternative approach for handling legislative acts, which maintains the ability to manage documents as hierarchical structures, with the creation of dedicated parthood graph relationships, but that enriches it with tools of additional utility. First, we demonstrate how our proposal seamlessly handles textual documents by focusing on one of the main issues of legislative knowledge management, that is, the law versioning problem, i.e., retrieving the text of a law in force in a certain timestamp, thus accounting for successive modifications and/or abrogations.

Then, we show how, by explicitly modelling references among acts as graph relationships, we enable more sophisticated tools to analyze and monitor trends in the legislative activity [10], such as by the development of intelligent systems that can *reason* over legislative data through the use of active rules in the form of Property Graph triggers [8]. To this aim, we develop a trigger-based warning system that monitors the legislation, capable of detecting harmful scenarios within the legislative system, such as the creation of legislative voids. We demonstrate how we capture the latter scenario by using active rules in the form of triggers expressed in Cypher, the closest language compliant with the Graph Query Language standard. We demonstrate the effectiveness of this warning system over the Italian legislation, for which we have implemented our PG-based schema and stored it within a Neo4j database. We experiment how warnings signaled by our reasoning-based approach significantly positively correlate with ex-post-legislative interventions that aim at fixing the legislative void, demonstrating their predictive power for a timely detection of legislative voids.

Finally, we analyze how association rules can be a useful

Published in the Proceedings of the Workshops of the EDBT/ICDT 2025 Joint Conference (March 25-28, 2025), Barcelona, Spain

*Corresponding author.

andrea1.colombo@polimi.it (A. Colombo); francesco.cambria@polimi.it (F. Cambria); francesco.invernici@polimi.it (F. Invernici)

0000-0002-7596-8863 (A. Colombo); 0009-0002-0268-4840 (F. Cambria); 0009-0002-5423-6978 (F. Invernici)

tool for knowledge discovery, providing insights into trends in the evolution of the legislative system. To this aim, by leveraging the recently developed association rule operator for property graphs [9], we present an application in which association rules can help in detecting the attitude of governments towards a topic of particular interest, i.e., whether there is discontinuity among successive governments. We focus on legislation related to "procurement", and we find that, in Italy, there is a continuous significant stream of creations and abrogations of rules governing this topic, certifying a certain degree of instability in the topic in recent years.

Overview. The remainder of the paper is organized as follows. Section 2 presents the Property Graph Schema that we adopt for modelling legislative knowledge, discussing its implementation in the Italian system and how it can retrieve time-dependent versions of laws. In Section 3, we present how tools like PG triggers and graph association rules can be used to facilitate knowledge discovery and automated monitoring systems. In Section 4, we perform our experiments to demonstrate the benefits of using a graph database and its rich set of tools to manage legislative knowledge.

2. Property Graphs for Legislative Data

Modelling legislative data and the interconnections between laws and articles in a property graph offers a structured and dynamic approach to understanding and managing the knowledge of a legislative system.

In this section, we present a property graph schema that aims to handle legislative data and discuss how we can leverage the expressiveness of a graph query language, such as Cypher, the closest GQL-compliant language, to handle a fundamental feature in legislative knowledge management, the retrieval of the correct version of a law based on a desired timestamp.

2.1. Property Graph Schema

In [11], we proposed the first approach for representing the Italian legislative system, in terms of documents and references, through a Property Graph, together with its schema [7]. While our proposal was country-specific, its foundation relies on an internationally adopted and OASIS standard [12], which aimed to define common elements that characterize legislative acts across multiple legislative traditions [13]. With no or little adaptations, our schema applies to multiple countries; thus, we can consider our schema powerful enough to represent any legislation.

In Figure 1, we visually present the schema. In a graph, nodes can represent laws connected to their articles (or, according to the legislative tradition, sections, clauses, etc.) through a parthood relationship. Metadata relevant to the law, such as title, publication date, and law domain (indicating the ministries responsible for the law's content), are assigned as properties directly to the law node, enabling a structured approach to query and analyze the legislative corpus. Articles are distinct nodes with properties that detail the specific sections of the law they represent. These properties include the article text, any headings or titles specific to the article, and a list of policy-related topics governed by that article. Attachments, which serve a different function

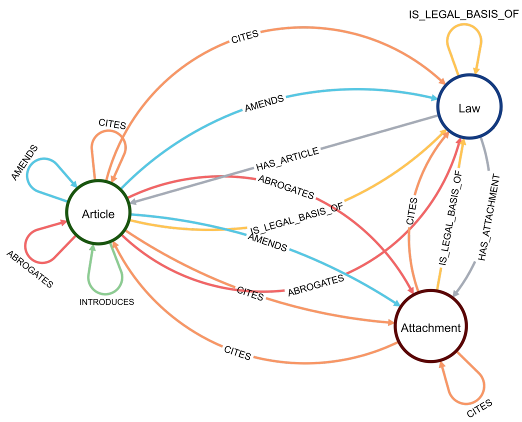

Figure 1: Graphical visualization of the Property Graph Schema. Properties are omitted for better visualization.

within a law document than articles, are also modelled as distinct nodes with their characteristics. Law, article and attachment nodes are interconnected using a set of directed edges that capture references, dependencies, and amendments that denote the evolution of a legislative system. We included four fundamental types of relationships which we identified as generic enough across multiple legislative traditions: an *"is the legal basis of"* dependence, an *amends*, an *abrogates* and a more generic *cites* reference.

The formal PG-Schema, as defined in [7], that we use to model legislative data is the following:

```
CREATE GRAPH TYPE lawsGraphType STRICT{
  (lawType: Law {id STRING, title STRING,
   lawNum INT, typeLaw STRING, publicationDate
   DATE, numArt INT, numAttach INT}),
  (articleType: Article {id STRING, title STRING,
   number INT}),
  (attachmentType: Attachment {id STRING, title
   STRING, type STRING}),
  (:lawType)-[hasArticleType: has_article]->
   (:articleType),
  (:lawType)-[hasAttachmentType:has_attachment]->
   (:attachmentType),
  (:lawType)-[referenceType: is_legal_basis_of]
   ->(:lawType),
  (:articleType)-[referenceType:is_legal_basis_of|
   amends|abrogates|cites]->(:lawType),
  (:articleType)-[referenceType: amends|abrogates|
   cites {paragraph STRING, newText STRING}]->
   (:articleType),
  (:articleType)-[referenceType: amends|abrogates
   {paragraph STRING, newText STRING}]->
   (:attachmentType),
  (:attachmentType)-[referenceType:
   is_legal_basis_of|cites]->(:lawType),
  (:attachmentType)-[referenceType: cites
   {paragraph STRING}]->(:articleType),
  (:attachmentType)-[referenceType: cites
   {paragraph STRING}]->(:attachmentType)}
```

Regarding edge properties, all defined types of references share the common attribute *paragraph*, which denotes, if applicable, the specific portion of the article's text that is being referenced. For instance, one could be interested in citing or amending only a specific portion of text within the article. Thus, such a feature can be captured by inserting the paragraph of interest. As a consequence of such a modelling choice, an *abrogates* edge whose *paragraph* property is NULL

Text of articles of the law of interest in their latest modified version before the desired timestamp

Text of articles of the law of interest, which have modifications only after the desired timestamp

Text of articles of the law of interest that have never been amended

In-force text of desired law at date T

```
CALL {
  MATCH (l:Law)-[:HAS_ARTICLE]->(a:Article)
  OPTIONAL MATCH (a)<-[r:ABROGATES|AMENDS|INTRODUCES]-(a2)<-[:HAS_ARTICLE]-(l2:Law)
  WHERE l2.publicationDate < datetime({$TIMESTAMP})
  WITH l.id as IDLAW, a.id as IDART, a.number as NUMART,  MAX(l2.publicationDate) AS LASTMOD
  WHERE IDLAW = {$IDLAW}
  WITH IDLAW, IDART, NUMART, LASTMOD
  MATCH (l:Law)-[:HAS_ARTICLE]->(a:Article)<-[r:ABROGATES|AMENDS|INTRODUCES]-(a2)<-[:HAS_ARTICLE]-(l2:Law)
  WHERE l.id = IDLAW AND a.id = IDART AND LASTMOD = l2.publicationDate
  RETURN IDLAW, IDART, r.newtext AS TEXT, NUMART

  UNION

  MATCH (l:Law)-[:HAS_ARTICLE]->(a:Article)<-[r:ABROGATES|AMENDS|INTRODUCES]-(a2)<-[:HAS_ARTICLE]-(l2:Law)
  WITH l.id AS IDLAW, a.id AS IDART, COUNT(r) AS NCHANGES
  WHERE IDLAW = {$IDLAW}
  WITH IDLAW, IDART, NCHANGES
  MATCH (l:Law)-[:HAS_ARTICLE]->(a:Article)<-[r:ABROGATES|AMENDS|INTRODUCES]-(a2)<-[:HAS_ARTICLE]-(l2:Law)
  WHERE a.id = IDART AND l2.publicationDate >= datetime({$TIMESTAMP})
  WITH IDLAW, IDART, a.number AS NUMART, a.text AS TEXT, NCHANGES, COUNT(*) AS FUTURECHANGES
  WHERE NCHANGES =  FUTURECHANGES
  RETURN IDLAW, IDART, TEXT, NUMART

  UNION

  MATCH  (l:Law)-[:HAS_ARTICLE]->(a:Article)
  WHERE NOT (a)<-[:ABROGATES|AMENDS|INTRODUCES]-() AND l.id = {$IDLAW}
  RETURN l.id AS IDLAW, a.id AS IDART, a.text AS TEXT, a.number AS NUMART
}
WITH TEXT, NUMART
WHERE TEXT IS NOT NULL
WITH TEXT, NUMART
ORDER BY NUMART ASC
RETURN COLLECT(TEXT)
```

Figure 2: Annotated Cypher query deriving the in-force text of law IDLAW at a certain TIMESTAMP. In red, the two variables to be set according to the user's requests. First, a subquery (within the CALL construct) is used to derive and union the law articles in-force at the desired timestamp. Then, a COLLECT function is used to produce the correct law text as a document.

implies a full abrogation of the destination article. A law becomes abrogated whenever all its articles have been abrogated. Thus, a law abrogation can be inferred by looking at whether all its articles have an ABROGATES edge without the *paragraph* property.[1]

Finally, *amends* and *abrogates* edges are the only types of references that also share a second property: *newText*. Such (textual) property captures the new text of the destination article node, as modified/abrogated by the source article node.

The Italian Legislative Property Graph We implemented our schema for the Italian legislation and derived a graph consisting of Italian national laws. The graph is stored in a Neo4j database (available at [14]), comprising 74k law nodes, 318k article nodes, 127k attachment nodes, 107k legal basis dependencies, 64k abrogations, 80k amendments, and 228k generic citations.

2.2. Managing Law Versioning via Graph Queries

Legislative systems evolve over time, generating a continuous stream of new laws that amend or repeal existing ones. Each change represents a new version of the law, capturing the updated text whenever a modification occurs. This process often results in an exponential increase in textual documents, as even minor changes necessitate the storage of additional files with the novel text.

By adopting the graph model and the schema that we proposed, we can seamlessly handle such feature by (i) storing the original textual version of each article of law within a node property, (ii) storing the modified textual version, as per amended by successive laws, as an edge property and (iii) delegating the computation of the in-force text for a law at a certain timestamp to a graph query, that, by leveraging publication dates available in law nodes, is capable to always infer the desired law version. By doing so, we avoid storing multiple textual versions of the same law, achieving a more efficient storage of legislative data and, at the same time, allowing a more structured and straightforward approach for querying statistics or detecting trends of a legislative system.

In Figure 2, we present the query in Cypher that we designed to manage law versioning based on our proposed schema. Such a query strongly relies on navigating graph patterns to combine temporal information. In detail, by specifying a TIMESTAMP T and a law of interest L (i.e., an IDLAW), we can get the text version of L at T by performing a union that concatenates articles of L in their latest modified version, as per modified by the law with the maximum publication date before T, with articles that have never been amended or have been amended only after T, thus do not have any incoming edge of types ABROGATES and AMENDS.

Finally, we note how this approach is replicable even on a more granular level, i.e., by considering paragraphs as graph nodes. The benefit would be even more pronounced since most of the modifications change only some words of a law, and a standard document versioning approach would require the storage of an additional document, even for tiny substitutions. Here, the challenge would be consistently modelling paragraph nodes and linking textual references to such nodes, since references do not always explicitly mention the paragraph, as we experienced with the Italian legislation.

3. Triggers and Association Rules

Triggers have been a fundamental feature since the inception of relational databases [15]; they were explored in depth in [16] and formalized in the ISO-ANSI SQL3 Standard [17]. Traditionally, triggers have been utilized to automate tasks that facilitate the development of automation systems [18]. In graph databases, adding reactive components has also been helpful in supporting important applications, such as

[1]In general, an abrogation can also directly repeal an entire law, which is also permitted by our schema. n the rest of the paper, we will not consider this case since it can be trivially considered by adding a UNION operator in all our applications.

the monitoring of pandemic events [8], which naturally are modeled into knowledge graphs stored in graph databases. Formalizations of triggers for graph database and Property Graph have recently been proposed [8, 19]. Due to the richness of the graph data model w.r.t. the relational model, triggers can also be leveraged to encode more complex (and interesting) events. For instance, since property graph items can be either nodes or relationships, triggers can include the creation and deletion of nodes/relationships as well as the setting and removal of their labels and properties. The strong analogy between triggers, i.e., active rules, and deductive rules, as defined in [20], has also enabled the use of triggers for adopting *reasoning* approaches over data modelled into knowledge graphs stored as property graphs.

Association rules (AR) mining was first introduced in [21, 22] and aim to uncover patterns or correlations within data. Classically, association rules take the form of $X \rightarrow Y$ where X is referred to as the body of the rule and Y as its head. Both body and head represent predicates or conditions that can be validated within the data, based on two key metrics: support and confidence. Association rules mining has been applied in many different domains: for instance, for highlighting which items are frequently purchased together [23, 24], for detecting connections between symptoms and diseases in medicine [25], or to uncover patterns of user interest in social networks [26]. Like triggers, the flexibility and richness of the graph data model allow association rule mining to uncover more insightful patterns within the data. The MINE GRAPH RULE operator, introduced in [9], leverages all the capabilities of the graph query language to simplify the definition and extraction of association rules from graph databases. While the operator is not fully integrated into the standard Cypher language, its syntax closely follows Cypher, ensuring clarity and readability. Furthermore, the queries used to extract association rules can be implemented in any graph database management system that supports GQL.

In the rest of this section, we present two practical applications of such tools, demonstrating their contribution to implementing more sophisticated monitoring activities in the legislative domain.

3.1. Warnings for Legal Basis Abrogation

In the legislative context, reasoning through active rules can be used to create automatic monitoring systems that derive warnings for anomalous events based on graph patterns. We identified an important application of such a feature in the monitoring events of repealing important acts that are *legal basis* to other pieces of legislation. Such abrogations might create legislative voids and hinder the applicability of third laws, which have their foundation in the repealed law. In other words, the content and rules stated in an act might depend on legislation abrogated by the newly enacted law.

In our schema, we modelled legislative dependencies through IS LEGAL BASIS OF edges, which define the foundations of a given law. Thus, to implement the application, we can use the graph structure to infer timely warnings about potentially harmful situations that must be monitored. Such warnings take the form of new special edges that can be visually inspected. In detail, we can define the following deductive rules that generate warnings for the considered problem:

1. Whenever a new law containing abrogations is published, we *compute* whether the new law has repealed an older law by leveraging the graph schema, as discussed in Section 2. In such cases, we *merge* REPEAL edges that connect the new law node with the abrogated law nodes.
2. By traversing the IS LEGAL BASIS OF edges, a second reasoning rule monitors whether the REPEAL of a law has created a legislative void in the legal foundation of a third in-force law. For such cases, it creates a warning edge informing about the potentially harmful situation (that we denote as an UNDERMINES edge).

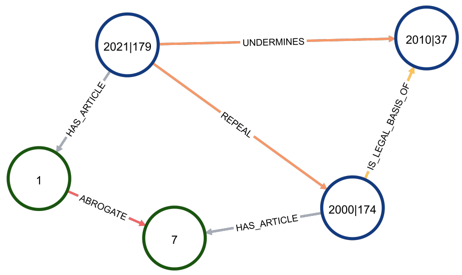

Figure 3: Law 2021/179 repeals law 2000/174, by abrogating its remaining in-force article (i.e., article 7). However, since law 2000/174 is a legal foundation of law 2010/37, law 2021/179 also undermines its validity since its foundation has been removed, raising a potential legislative void.

A visual example of the reasoning process is available in Figure 3.

The first rule can be encoded within the following trigger expressed in Cypher, a possible implementation of PG-Triggers, as defined in [8]:

```
UNWIND $createdRelationship AS newRel
MATCH p=(l:Law)-[:HAS_ARTICLE]->(a:Article)
    <-[newRel:ABROGATES]-(a2:Article)
    <-[:HAS_ARTICLE]-(newLaw:Law)
WHERE newRel.paragraph IS NULL
WITH l, a, newLaw
MATCH (l)-[:HAS_ARTICLE]->(a)<-[r:ABROGATES]-()
WHERE r.paragraph IS NULL
WITH l, l.numArt AS NUMART, COUNT(DISTINCT a)
    AS NUMREPEALS, newLaw
WHERE NUMREPEALS >= NUMART
MERGE (newLaw)-[r:REPEAL]->(l)
```

where the UNWIND clause captures, within the Neo4j ecosystem, the insertion of a new ABROGATES edges which derive from the insertion of a new law node. If a graph pattern denoting an article abrogation without the *paragraph* property exists (denoting, according to our schema, a full abrogation), then the full abrogation is tested by monitoring whether all its articles have been repealed. In such a scenario, the trigger *merges* a new edge denoting an abrogation (i.e., s REPEAL) of the destination law node.

The second rule can be implemented as:

```
UNWIND $createdRelationships AS newRel
MATCH (a)<-[:HAS_ARTICLE]-(newLaw:Law)
    -[newRel:REPEAL]->(l:Law)-[:IS_LEGAL_BASIS_OF]
    ->(l2:Law)-[:HAS_ARTICLE]->(a2:Article)
WHERE NOT EXISTS (()-[:REPEAL]->(l2))
AND newLaw.id <> l2.id
AND NOT EXISTS ((a)-[:ABROGATES|CITES|AMENDS|
    INTRODUCES]->(a2))
MERGE (newLaw)-[:UNDERMINES]->(l2)
```

where, here, the UNWIND clause captures the newly created REPEAL edges and checks whether it exists a non-abrogated law whose the repealed one is the legal basis of. Within the WHERE conditions, we exclude the cases in which the repealing law already contains any explicit "correction" to the destination law and its articles. If such a pattern exists, we merge new derived relationships, i.e., UNDERMINES edges, which denote a potential legislative void since the newly published law is deleting a law that is used as a legal foundation in a third law. Thus, such edges require attention from the legislator and any interested stakeholders in the domain of the law. The warnings might translate into corrective actions, i.e., new laws or articles that fix legislative voids that might have been generated.

3.2. Patterns of Government Attitude

In a legislative graph proposed in Section 2, items represent laws, articles, and attachments. Such a schema can be easily enriched by adding governments and topic nodes[2]. The former represents the governments under which laws have been enacted; the latter connects each legislative node to the topics its content refers to. With these enrichments, we can exploit (graph) association rules to understand which and if governments have the attitude to profoundly change legislation referring to a topic for which its predecessor government had also previously implemented legislation. Through this application, we can detect *temporal graph patterns* that monitor the behaviour of governments and identify discontinuity evolution of the legislation. The mining of this kind of pattern can be achieved by adopting the MINE GRAPH RULE operator, whose syntax was proposed in [9], and can be written as:

```
MINE GRAPH RULE FrequentChangesOnTopic
GROUPING ON (t:Topic)
WHERE t.name CONTAINS '$topicName'
DEFINING
    BODY AS (t)<-[:OF_TOPIC]-(a:Article)
        <-[:HAS_ARTICLE]-(l:Law)<-[:CREATED_UNDER]
        -(g:Government)-[:SUCCEDED_BY]
        ->(g2.Government)
    HEAD AS (t)<-[:OF_TOPIC]-(a2:Article)
        <-[:ABROGATES]-(a3:Article)<-[:HAS_ARTICLE]
        -(l2:Law)<-[:CREATED_UNDER]-(g3:Government)
WHERE g2 = g3
IGNORE a, l, g2, a2, a3, l2
EXTRACTING RULES WITH SUPPORT > 0.2
               AND CONFIDENCE > 0.6
```

which extracts whether *a certain topic of interest, regulated under a certain government, is also significantly reshaped by its direct successor in government.* Therefore, the output of such a rule is statistically significant patterns of discontinuity that allow us to monitor a possible important shift of regulation from one government to another regarding a certain topic. We highlight how such output is more high-level than looking for modifications and abrogations that a government made to the legislation implemented by its predecessor. In fact, the complexity of the legislative system hinders the result of such a query since these shifts could also be captured indirectly through third laws, which share the same (or similar) topic.

[2]Multiple approaches can be used to derive topics from textual documents, either semi-supervised ones [27] or state-of-the-art LLM-based topic extraction tools [28]

Repealing Law	Warnings	Legislative Interventions
1949/264	1	4
1954/615	1	0
1967/18	49	20
1967/601	9	3
...	...	...

Table 1
Sample of the warnings generated by a repealing law and the number of legislative interventions that have occurred, within one year after the publication of the repealing law, to third laws having as foundation a law abrogated by the "repealing" law.

4. Implementation and Results

In this section, we execute the proposed triggers and association rules within the Italian legislation that we have already modelled in a property graph [14], aiming to demonstrate the efficacy and utility of such tools in a real-world scenario. In particular, we show how warnings generated by our triggers significantly correlate with ex-post interventions, i.e., correction measures that modify or add details about the law signalled by the alert system based on triggers. Then, we implement an AR in the Italian graph to analyze whether we can monitor and find patterns of the legislation related to procurement rules, which are under strong scrutiny in the daily Italian news. We discover that in recent years, many governments have had a reshaping attitude toward such legislation, with profound changes and discontinuity among recent governments, which harm the stability of legislation, one the drivers of economic growth [29].

4.1. Relevance of Warnings

As discussed in Section 3.1, we consider as a warning the emergence of an UNDERMINES edge, derived via reasoning with triggers. We implemented the triggers in a Neo4j database, which we also used to store the graph of the Italian legislation. To implement triggers as reasoning rules, capable of inferring novel knowledge through deductions, as required by this use case, we made use of a publicly available Neo4j plugin[3] that, through the use of a trigger controller, manages the correct execution order of triggers that enables reasoning.

To test the significance of these warnings, we adopted a historical analysis strategy: we replicated the (ordered) insertion of each law within the graph to let the Neo4j triggers activate based on the legislative situation at each timestamp.

First, we computed the number of warnings generated for each law, which we depict in Figure 4. Then, we correlated such values with the number of ex-post interventions that the legislator had to make within the year after the publication of the law to fix or repair a legislative void. We count interventions as the distinct number of references to the "undermining" law that occurs in the following 365 days after the official publication. Thus, for each law that activates the triggers, we get two values: the number of warnings generated and the number of interventions. In Table 1, we present a portion of the two values that we computed for laws that have been signalled as warnings.

Finally, we compute the Pearson correlation coefficient to measure the linear relationship between the two datasets.

[3]The implementation is available at [30]

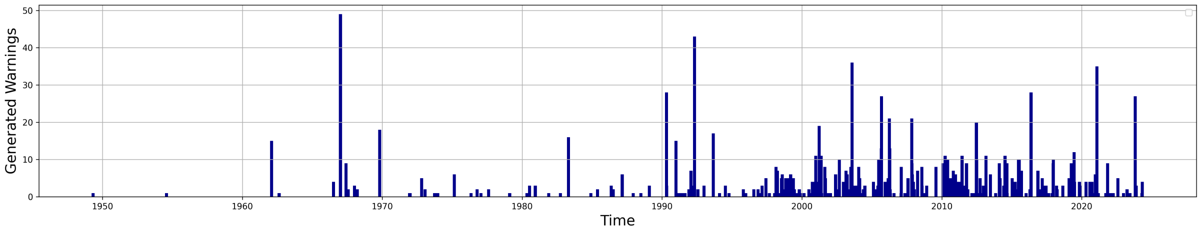

Figure 4: Warnings generated by our trigger-based reasoning system, aiming to monitor the abrogations of relevant laws. Each bar represents the number of warnings generated whenever a new law is published, i.e., the number of UNDERMINES relationships that are merged within the graph.

The idea is that a significant positive correlation coefficient implies that the higher the number of warnings generated, the higher the probability of necessary intervention to fix the legislative void, giving our warning system also some predictive power to predict harmful legislation. Since the distributions of warnings and interventions are not normally distributed, we adopted a bootstrap approach [31, 32] to estimate the sampling distribution of the correlation coefficient and to test whether it is significantly different from zero. We observed a Pearson coefficient of 0.394 and a p-value of 0.02, implying a significant positive relationship between the two series, which can be interpreted as a higher probability of intervention whenever more warnings are generated, i.e., a "central" law is repealed and requires more interventions.

4.2. Evolution of Procurement Regulation

As a representative example, we identified regulation regarding "procurement" as an interesting topic to analyze through association rules. The theme is under scrutiny by the general public for its long-standing episodes of corruption in the country, as also studied in dedicated economic research [33]. Through the running of our association rule tool for this topic, we aim to monitor the behaviour of governments in continuously modifying legislation in this area.

We implemented the MINE GRAPH RULE operator in Neo4j[4] a previous work (see [9]). The output of the AR run for the procurement legislation is presented in Table 2. Considering the number of topics in the graph, setting the support and confidence thresholds to 0.2 effectively identifies strong rules. Out of the 69 governments of the Italian Republic, 16 have significantly reshaped the topics their predecessor had worked on. An alarming trend is that this "practice" has intensified in recent years. Notably, 15 out of the 16 governments identified began after 1990. Another concerning behavior highlighted by the extracted rules is that governments succeeding those of a different political alignment are more likely to repeal pieces of procurement legislation, suggesting a dismantling of the predecessors' work. For instance, one of the rules with the highest support and confidence values reveals that the fourth Berlusconi government (right-wing) reshaped procurement legislation previously modified or enacted by the second Prodi government (left-wing).

Head Government	**Body** Predecessor	**Support**	**Confidence**
I Meloni	I Draghi	0.57	0.80
II Conte	I Conte	0.29	0.66
I Conte	I Gentiloni	0.57	1.00
I Gentiloni	I Renzi	0.57	0.80
I Renzi	I Letta	0.28	1.00
I Letta	I Monti	0.28	0.66
I Monti	IV Berlusconi	0.57	0.80
IV Berlusconi	II Prodi	0.57	1
II Prodi	III Berlusconi	0.43	0.5
II Berlusconi	II Amato	0.29	1
II D'Alema	I D'Alema	0.29	0.33
I D'Alema	I Prodi	0.29	0.5
I Prodi	I Dini	0.29	0.5
I Ciampi	I Amato	0.29	1
VII Andreotti	VI Andreotti	0.43	0.75
III Moro	II Moro	0.43	1

Table 2
Output of the MINE GRAPH RULE operator used to discover patterns of changes in procurement regulation. Results have been ordered by the temporal dimension, i.e., starting from the most recent pair of adjacent governments, and depict a scenario in which the government in the body column enacted relevant legislation about "procurement" which was profoundly reshaped by the (successor) government in the head column.

5. Conclusions

In this paper, we built upon recent efforts in modelling legislative systems within graph databases, describing the motivations and benefits of adopting Property Graphs as a powerful solution for comprehensive legislative knowledge management. We demonstrated how adopting a graph database and our schema can seamlessly handle the temporal aspects and evolution of legislative acts, delegating to queries the task of retrieving the correct information, such as computing the in-force text, instead of storing multiple versions of the same act. Then, we looked at recent tools recently adopted and developed by the Property Graph community, i.e., triggers and association rules, to offer innovative and powerful applications that can be useful in conducting advanced legislative knowledge management. To this aim, we demonstrated how triggers can be used to build a simple but effective warning system that monitors whether legislative voids are created. Finally, we used association rules to discover significant patterns in the evolution of the Italian legislation referred to procurement, discovering an increasing trend of instability in such legislation. In future work, we will further explore the utility of these tools for knowledge discovery, refining their applications in the legislative domain with the help of experts and identifying new domains that could benefit from these advancements.

[4]The implementation is available at [34]

References

[1] C. Lupo, F. Vitali, E. Francesconi, M. Palmirani, R. Winkels, E. de Maat, A. Boer, P. Mascellani, et al., General xml format (s) for legal sources., Technical report, IST-2004-027655 ESTRELLA European project for Standardised Transparent Representations in order to Extend Legal Accessibility (2007).

[2] R. Hoekstra, J. Breuker, M. Di Bello, A. Boer, et al., The lkif core ontology of basic legal concepts., CEUR Workshop Proceedings 321 (2007) 43–63. 2nd Workshop on Legal Ontologies and Artificial Intelligence Techniques, LOAIT 2007 ; Conference date: 04-06-2007 Through 04-06-2007.

[3] T. Athan, H. Boley, G. Governatori, M. Palmirani, A. Paschke, A. Wyner, Oasis legalruleml, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law, ICAIL '13, Association for Computing Machinery, New York, NY, USA, 2013, p. 3–12.

[4] G. Barabucci, L. Cervone, M. Palmirani, S. Peroni, F. Vitali, Multi-layer markup and ontological structures in akoma ntoso, in: International Workshop on AI Approaches to the Complexity of Legal Systems, Springer, Springer, Beijing, China, 2009, pp. 133–149. doi:10.1007/978-3-642-16524-5_9.

[5] W. Linder, Swiss legislation in the era of globalisation: A quantitative assessment of federal legislation (1983-2007), Swiss Political Science Review 20 (2014) 223–231. doi:https://doi.org/10.1111/spsr.12100.

[6] A. Bonifati, D. Braga, A. Campi, S. Ceri, Active xquery, in: Proceedings 18th International Conference on Data Engineering, 2002, pp. 403–412. doi:10.1109/ICDE.2002.994754.

[7] R. Angles, A. Bonifati, S. Dumbrava, G. Fletcher, A. Green, J. Hidders, B. Li, L. Libkin, V. Marsault, W. Martens, F. Murlak, S. Plantikow, O. Savkovic, M. Schmidt, J. Sequeda, S. Staworko, D. Tomaszuk, H. Voigt, D. Vrgoc, M. Wu, D. Zivkovic, Pg-schema: Schemas for property graphs, Proc. ACM Manag. Data 1 (2023). URL: https://doi.org/10.1145/3589778. doi:10.1145/3589778.

[8] S. Ceri, A. Bernasconi, A. Gagliardi, D. Martinenghi, L. Bellomarini, D. Magnanimi, Pg-triggers: Triggers for property graphs, in: Companion of the 2024 International Conference on Management of Data, SIGMOD/PODS '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 373–385. URL: https://doi.org/10.1145/3626246.3653386. doi:10.1145/3626246.3653386.

[9] F. Cambria, F. Invernici, A. Bernasconi, S. Ceri, Mine graph rule: A new cypher-like operator for mining association rules on property graphs, arXiv preprint arXiv:2406.19106 (2024).

[10] A. Colombo, Leveraging knowledge graphs and llms to support and monitor legislative systems, in: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, CIKM '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 5443–5446. URL: https://doi.org/10.1145/3627673.3680268. doi:10.1145/3627673.3680268.

[11] A. Colombo, A. Bernasconi, S. Ceri, An llm-assisted etl pipeline to build a high-quality knowledge graph of the italian legislation, Information Processing and Management 62 (2025) 104082. URL: https://www.sciencedirect.com/science/article/pii/S030645732500024X. doi:https://doi.org/10.1016/j.ipm.2025.104082.

[12] OASIS, Akoma ntoso version 1.0 becomes an oasis standard, 2018. Available online at: https://www.oasis-open.org/news/announcements/akoma-ntoso-version-1-0-becomes-an-oasis-standard/, last accessed on 01.05.2024.

[13] F. Vitali, M. Palmirani, et al., Akoma ntoso: flexibility and customization to meet different legal traditions, BALISAGE SERIES ON MARKUP TECHNOLOGIES 24 (2019) 1–9.

[14] A. Colombo, Knowledge graph of the italian legislation, 2024. URL: https://zenodo.org/doi/10.5281/zenodo.13798158. doi:10.5281/ZENODO.13798158.

[15] K. P. Eswaran, Aspects of a trigger subsystem in an integrated database system, in: Proceedings of the 2nd International Conference on Software Engineering, ICSE '76, IEEE Computer Society Press, Washington, DC, USA, 1976, p. 243–250.

[16] J. Widom, S. Ceri, Active database systems: Triggers and rules for advanced database processing, Morgan Kaufmann, 1995.

[17] J. Melton, A. R. Simon, SQL: 1999: understanding relational language components, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[18] S. Schmitz, M. Schluetter, U. Epple, Automation of automation — definition, components and challenges, in: 2009 IEEE Conference on Emerging Technologies and Factory Automation, 2009, pp. 1–7. doi:10.1109/ETFA.2009.5347197.

[19] L. Bellomarini, A. Bernasconi, S. Ceri, A. Gagliardi, D. Magnanimi, D. Martinenghi, et al., Towards a standard for triggers in property graphs, in: CEUR WORKSHOP PROCEEDINGS, volume 3741, CEUR-WS, 2024, pp. 70–79.

[20] S. Ceri, J. Widom, Deriving incremental production rules for deductive data, Information Systems 19 (1994) 467–490. URL: https://www.sciencedirect.com/science/article/pii/0306437994900027. doi:https://doi.org/10.1016/0306-4379(94)90002-7.

[21] R. Agrawal, T. Imieliński, A. Swami, Mining association rules between sets of items in large databases, in: Proceedings of the 1993 ACM SIGMOD international conference on Management of data, 1993, pp. 207–216.

[22] R. Srikant, R. Agrawal, Mining generalized association rules, Future generation computer systems 13 (1997) 161–180.

[23] S. W. Changchien, T.-C. Lu, Mining association rules procedure to support on-line recommendation by customers and products fragmentation, Expert systems with applications 20 (2001) 325–335.

[24] T. Osadchiy, I. Poliakov, P. Olivier, M. Rowland, E. Foster, Recommender system based on pairwise association rules, Expert Systems with Applications 115 (2019) 535–542.

[25] M. Tandan, Y. Acharya, S. Pokharel, M. Timilsina, Discovering symptom patterns of covid-19 patients using association rule mining, Computers in biology and medicine 131 (2021) 104249.

[26] H. Si, J. Zhou, Z. Chen, J. Wan, N. N. Xiong, W. Zhang, A. V. Vasilakos, Association rules mining among interests and applications for users on social networks, IEEE Access 7 (2019) 116014–116026.

[27] A. Bernasconi, F. Invernici, S. Ceri, et al., Tetys: Towards the next-generation open-source web topic explorer, in: CEUR WORKSHOP PROCEEDINGS, volume 3692, CEUR-WS, 2024, pp. 26–33.

[28] R. Y. Maragheh, C. Fang, C. C. Irugu, P. Parikh, J. Cho, J. Xu, S. Sukumar, M. Patel, E. Korpeoglu, S. Kumar, et al., Llm-take: theme-aware keyword extraction using large language models, in: 2023 IEEE International Conference on Big Data (BigData), IEEE, 2023, pp. 4318–4324.

[29] K. W. Dam, The law-growth nexus: The rule of law and economic development, Rowman & Littlefield, 2007.

[30] D. Magnanimi, Neo4j APOC light-weight reasoner, https://github.com/davidemagnanimi/apoc, 2023.

[31] B. Efron, Bootstrap Methods: Another Look at the Jackknife, Springer New York, New York, NY, 1992, pp. 569–593. URL: https://doi.org/10.1007/978-1-4612-4380-9_41. doi:10.1007/978-1-4612-4380-9_41.

[32] J. L. Rasmussen, Estimating correlation coefficients: Bootstrap and parametric approaches., Psychological Bulletin 101 (1987) 136.

[33] F. Decarolis, R. Fisman, P. Pinotti, S. Vannutelli, Rules, discretion, and corruption in procurement: Evidence from italian government contracting, Technical Report, National Bureau of Economic Research, 2020.

[34] F. Invernici, F. Cambria, Neo4j Mine Graph Rule Operator, https://github.com/FrInve/mine_graph_rule, 2024.