



**Fernando Antonio Dantas Gomes Pinto**

**Compliance Reasoning on Legal Norms: a  
logic-based approach**

**Tese de Doutorado**

Thesis presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Ciências – Informática.

Advisor: Prof. Edward Hermann Haeusler

Rio de Janeiro  
April 2024



**Fernando Antonio Dantas Gomes Pinto**

**Compliance Reasoning on Legal Norms: a  
logic-based approach**

Thesis presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Ciências – Informática. Approved by the Examination Committee:

**Prof. Edward Hermann Haeusler**

Advisor

Departamento de Informática – PUC-Rio

**Prof. Sérgio Lifschitz**

Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

**Profa. Fernanda Araujo Baião**

Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio

**Prof. Altigran Soares da Silva**

UFAM

**Prof. Bruno Lopes Vieira**

UFF

Rio de Janeiro, April 26th, 2024

All rights reserved.

**Fernando Antonio Dantas Gomes Pinto**

Has a Bachelor's Degree in Informatics from Cesmac (Alagoas) and an MSc in Computational Knowledge Modeling at the Federal University of Alagoas (UFAL) as well, specializing in Theory of Computation and Logic, more specifically in SAT Solvers and description logic.

Bibliographic data

Pinto, Fernando Antonio Dantas Gomes

Compliance Reasoning on Legal Norms: a logic-based approach / Fernando Antonio Dantas Gomes Pinto; advisor: Edward Hermann Haeusler. – 2024.

116 f: il. color. ; 30 cm

Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2024.

Inclui bibliografia

1. Informática – Teses. 2. Auditoria Legal. 3. Lógica Descritiva. 4. iALC. 5. Raciocínio Legal. 6. SAT Solver. I. Hermann Haeusler, Edward. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

## Acknowledgments

I would like to express my deep gratitude to everyone who contributed to completing this thesis.

Firstly, I thank my family. To my wife, Simone, and my children, Fernando and Sofia, for their patience, understanding and unconditional love. This was a difficult period, as I was away from you for some time, and I thank you for all your support, even from a distance. To my mother, Mary Stella, and my sisters, Aurelina and Juliana, for always believing in me and encouraging me to move forward, even when my presence at family events was sacrificed in favor of my dedication to my doctorate.

To my advisor, prof. Edward Hermann Haeusler, I express my immense gratitude for all the teaching, patience and friendship. I am grateful for the valuable moments we spent together, from the coffees where we exchanged ideas to the crucial guidance fundamental to completing this work.

To my friends at TecMF, I would like to thank you for the good times we shared in our weekly meetings. During this challenging journey, your company and camaraderie were essential to maintaining motivation and balance.

To all of you, my sincere thanks.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior — Brasil (CAPES) — Finance Code 001.

## Abstract

Pinto, Fernando Antonio Dantas Gomes; Hermann Haeusler, Edward (Advisor). **Compliance Reasoning on Legal Norms: a logic-based approach**. Rio de Janeiro, 2024. 116p. Doctoral Thesis – Departamento of Informatics, Pontifícia Universidade Católica do Rio de Janeiro.

Ensuring that a knowledge base with public administration acts contains only facts in accordance with its legislation becomes a challenge for any public manager. To achieve this, given the large volume of data generated by public companies, it is necessary to apply technological resources that assist in the process of analyzing the compliance of these acts. This work presents a computational architecture capable of extracting information published in official gazettes and then serializing it into two knowledge bases, RDF/XML triples of facts and RDF/XML triples of rules formalized in *iALC* logic, an intuitionistic description logic. To ensure the consistency of this knowledge base, a SAT Solver for *iALC* was developed in the form of an intuitionistic semantic tableau. An extension of the first-order intuitionist tableau presented by Fitting (1960). This SAT Solver is part of a module that generates models and counter-examples for rules formalized in *iALC* and generates a preliminary query code in SPARQL. This approach allows infer and certify the quality of the data available in the RDF/XML knowledge base of facts. To guarantee the quality of our SAT Solver, we carry out the soundness proof of its rules. To ensure the quality of our logical approach, we built a set of 21 Competency Questions and applied our tool. The results of this case study showed our approach's effectiveness and efficiency.

## Keywords

Legal auditing; Description Logic; *iALC*; Legal Reasoning; SAT Solver.

## Resumo

Pinto, Fernando Antonio Dantas Gomes; Hermann Haeusler, Edward. **Raciocínio de Compliance sobre Normas Legais: uma abordagem baseada em lógica**. Rio de Janeiro, 2024. 116p. Tese de Doutorado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Garantir que uma base de conhecimento com atos da administração pública contenha apenas fatos em conformidade com sua legislação torna-se um desafio para qualquer gestor público. Para isso, dado o grande volume de dados gerados por empresas públicas, faz-se necessário o emprego de recursos tecnológicos que auxiliem o processo de análise de conformidade destes atos. Este trabalho apresenta uma arquitetura computacional capaz de extrair informações publicados dos diários oficiais e então serializá-los em duas bases de conhecimento: triplas RDF/XML de fatos e triplas de RDF/XML de regras formalizadas em lógica *iALC*, uma lógica de descrição intuicionista. Para garantir a consistência desta base de conhecimento, foi desenvolvido um *SAT Solver* para *iALC* em forma de tableau semântico intuicionista. Uma extensão do tableau intuicionista de primeira ordem apresentado por Fitting (1969). Este *SAT Solver* faz parte de um módulo que além de gerar modelos e contra-exemplos para as regras formalizadas em *iALC*, também gera um código preliminar de consultas em SPARQL. Esta abordagem permite inferir e certificar a qualidade dos dados disponíveis na base de conhecimento RDF/XML de fatos. Para garantir a qualidade do nosso *SAT Solver*, fizemos a prova de *soundness* das suas regras. Para garantir a qualidade da nossa abordagem lógica, construímos um conjunto de 21 Questões de Competência e aplicamos à nossa ferramenta. Os resultados deste estudo de caso mostraram a eficácia e eficiência da nossa abordagem.

## Palavras-chave

Auditoria Legal; Lógica Descritiva; *iALC*; Raciocínio Legal; SAT Solver.

## Table of contents

<b>1</b>	<b>Introduction</b>	<b>11</b>
1.1	A framework vision	15
1.2	The Formal Definition of the Framework	16
1.3	Computational Complexity	18
1.4	Abstraction of nonconformity detection	18
1.5	Previous Results of Architecture	19
1.6	Organization of Chapters	20
<b>2</b>	<b>Architectural Design for Reasoning</b>	<b>21</b>
2.1	Related Works	21
2.2	Architectural Design	23
<b>3</b>	<b>Legal Extraction and Normalization Module (ENM)</b>	<b>25</b>
3.1	Well-Formatted Document	25
3.1.1	Syntactic structure of the law	27
3.2	Lexical and syntactic analysis	28
3.3	Related Works	28
3.4	Official Gazette Grammar	29
3.5	Official Gazette Patterns	31
3.6	ENM Modules	31
3.6.1	The Encoding Process	33
3.6.2	The Audit File	34
3.7	Evaluation and Results for ENM	35
3.7.1	Scope	35
3.7.2	Results	36
3.8	An Alternative with Machine Learning	38
3.8.1	Results	39
<b>4</b>	<b>Formula Generator Module (FGM)</b>	<b>41</b>
4.0.1	FGM Modules	43
4.1	From Law Text to ASTs	44
<b>5</b>	<b>Knowledge Base Reasoner Module (KRM)</b>	<b>47</b>
5.1	KRM Modules	47
5.2	Logical-base components	48
5.3	The Tableau System for <i>iALC</i>	52
5.3.1	Related works	56
5.3.2	iALC Tableau Rules	56
5.4	Main Properties	58
5.4.1	Soundness	58
5.5	Reasoning in <i>iALC</i> tableau calculus	62
5.6	The KRM Machine	63
5.6.1	The KRM Kernel	64
5.6.2	The Model and Counter-model Logs File	67

5.6.3	From Counter-model to SPARQL	68
5.6.4	The KRM Solver Application	73
<b>6</b>	<b>Evaluation and Results</b>	<b>76</b>
6.1	Reasoner Evaluation	76
6.1.1	Scoping	77
6.1.2	Planning	77
6.1.3	Operation	80
6.1.4	Analysis and Interpretation	88
<b>7</b>	<b>Conclusion and Future Works</b>	<b>90</b>
<b>8</b>	<b>Appendix</b>	<b>103</b>



## List of tables

Table 3.1	Regular expression pattern for the employee “Nomear” act.	32
Table 3.2	Result between the two techniques.	39
Table 3.3	Preprocessing experiments time.	39
Table 5.1	Some general guidelines for transforming <i>iALC</i> to a SPARQL query.	72
Table 6.1	Summary of correct and incorrect CQs.	88
Table 6.2	Query coverage summary.	88
Table 6.3	KRM SAT Solver Accuracy.	89
Table 6.4	KRM Query Coverage.	89

*Truth is the daughter of time,  
not of authority.*

**Francis Bacon**, *Novum Organum*.

# 1

## Introduction

Law, in Latin *legere*, means “that which is read”, is a set of legal rules (also known as “legal norms”) constructed by a legislative entity. These legal norms aim to impose a series of rules that guide our behavior in social, economic or political activities. When this obligation is not practiced, the company or citizen may have penalties or sanctions.

Within our legislation, there is a set of laws that regulate and control public activity. Here, this concept of regulation and control can be interpreted as a set of laws that govern public administration itself. Each federative entity (municipal or state) has its own set of laws and this capacity for self-regulation makes public companies environments of complex inspection. For the public manager, the problem resides precisely in this complexity in supervising the application of his acts, if they are in accordance with the law. Often, the excess of laws and the large number of public acts make it difficult to comply with legislation. But then, in an environment with so many laws and legal relationships, how can non-compliance be monitored?

In this sense, we can say that effective public management requires the implementation of public management computational mechanisms that help the compliance of its processes. In this work, the concept of compliance is directly linked to the aspects of complying with laws, and our objective is to prevent the public manager from committing crimes against the public administration. Consequences for non-compliance include fines, sanctions, and lawsuits.

The auditing is an important activity that aims to help organizations improve their internal processes. Unlike private companies, public auditing is governed by legal instruments that try to minimize the impact of non-compliance with legislation.

Based on what has been said, the government has been trying to implement for new audit models in public companies. Unlike carrying out the audit itself, it transfers this activity to society. An example of this is the participation of society in government decisions based on the analysis of public data available on the Internet. Social participation in accessing these data was already foreseen since the 1988 constitution, but limited to the Information

Communication Technology (ICT) tools available at that epoch.

According to the Federative Constitution of Brazil [1], in its Art. 5, item *XXXIII*, it is said that every public department is obliged to provide information generated by its activities to the individual with particular or collective interest. This law punishes the manager who does not comply with such access within the deadlines required by this law.

Only in 2011, given the great social demand for public transparency, did the federal government draft the Access to Information Law (LAI) [2], causing debates on the treatment and how to make this information accessible on the Web.

Even with all the government's efforts to make information available, we know that a large amount of information is not available for online access. To minimize this limitation, many governments have published a large list of official gazettes in digital format how another way to promote public transparency.

Official Gazette is a legal document for use by the administration to make its actions transparent to society, making its actions public. According to Art. 37 of the Constitution, the Brazilian public administration is supported by 5 legal principles: **Legality** - all public acts must be guided by the law. **Impersonality** - personal interests cannot override the public interest, and state power cannot be used for personal gain either. **Morality** - Public servants and other State workers must follow ethical and moral standards. **Publicity** - *all acts of public administration must be done with the knowledge of the population*, that is, they must be publicized so that everyone is aware. Public documents need to be accessible to everyone. **Efficiency** - the service provided by the public administration needs to be efficient, having the best result at the lowest possible cost and in the fastest way, always aiming at quality.

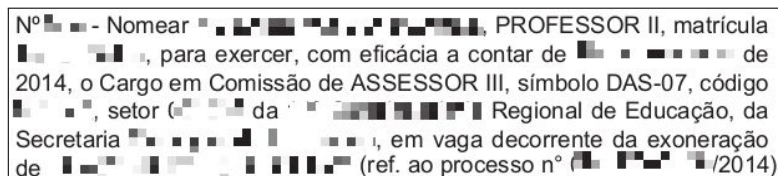
Thus, given what has been presented, we chose the official gazette as the main source of public data. All laws and legal statements are available across the various sections of this document.

During our research, we observed that the official gazette consistently exhibits a well-structured format across its various sections, adhering to a consistent writing pattern. We found that this characteristic is supported by a regulation that defines formatting rules for the construction of laws. Indeed, a Brazilian normative text can be viewed as a well-formed structural publication, akin to a template. This structure, which is evident, resembles a formal syntactic structure. An attempt to standardize legal texts can be seen in laws [3] and [4]. This standard mandates that public managers, when

publishing an act, must write the text following this template.

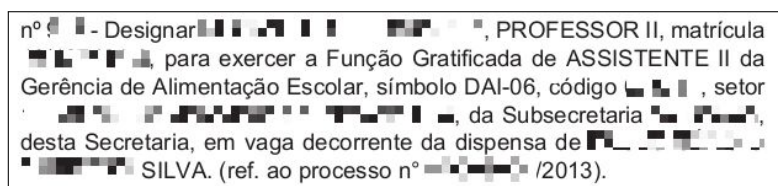
For this work, a Statement is any administrative act document from any public authority, containing instructions regarding the application of laws such as service execution standards, appointments, dismissals, punishments, or any other determination within its competence. We will analyze its several syntactic structures from a formal perspective. We will try to capture the propositions present in the text of the law and its correct application in accordance with the legal act. According to [5, p. 4], the legal norm works as an interpretation scheme, that an act of human conduct, the result of a normative interpretation, constitutes a **legal act that can be valid or invalid**.

To illustrate what is presented, Figs. 1.1 and 1.2 are cases extracted from an official gazette (hypothetical example) where a legal statement does not comply with the legal norm. In this case, a public servant could not be in a commissioned position (“ASSESSOR III”) and a gratified function (“ASSISTENTE II”) simultaneously. In this case, there was no exoneration for these positions. Basically, the objective is to identify these cases of non-compliance when they are published in the Official Gazette.



Nº [redacted] - Nomear [redacted], PROFESSOR II, matrícula [redacted], para exercer, com eficácia a contar de [redacted] de 2014, o Cargo em Comissão de ASSESSOR III, símbolo DAS-07, código [redacted], setor ([redacted] da [redacted] Regional de Educação, da Secretaria [redacted], em vaga decorrente da exoneração de [redacted] (ref. ao processo nº [redacted] /2014)

Figure 1.1: Legal statement for a commissioned position.



nº [redacted] - Designar [redacted], PROFESSOR II, matrícula [redacted], para exercer a Função Gratificada de ASSISTENTE II da Gerência de Alimentação Escolar, símbolo DAI-06, código [redacted], setor [redacted], da Subsecretaria [redacted], desta Secretaria, em vaga decorrente da dispensa de [redacted] SILVA. (ref. ao processo nº [redacted] /2013).

Figure 1.2: Legal statement for a gratified function.

When improperly holding multiple public positions, a public employee faces not only disciplinary proceedings that may lead to dismissal but also may be subject to judicial proceedings for administrative misconduct. Administrative misconduct entails illegal acts committed by public officials while performing their duties within the Brazilian public administration.

Ensuring that a knowledge base with public administration acts only contains acts in accordance with its legislation becomes a challenge for any

public manager. To achieve this, given the large volume of data, it is necessary to employ technological resources that assist in this compliance process.

Hence, **the objective of this research is to define a logical architecture capable of extracting legal representations from the Official Gazette and describing them as formulas of the intuitionistic descriptive logic (*iALC*), as well as making inferences capable of looking at situations of non-compliance with the law, represented by the subset of legal norms of interest.**

This activity of inferring the validity of legal statements means verifying their subordinate relationship with the laws that govern them. As presented in the introduction, this type of analysis is not a trivial activity. In the more traditional settings, the literature reports the use of deontic logic to formalize the normative aspects of legal knowledge. In previous works [6], the authors showed how *iALC* logic avoids some Contrary to Duty paradoxes, such as Chisholm's paradox, the good Samaritan, among others, when the law is formalized in deontic logic.

An important concept in [6] is Valid Legal Statement (VLS) as a single norm (regulate something). However, we recommend [7] for a better explanation.

Figures 1.3 and 1.4 promote a general understanding of our proposed architecture.

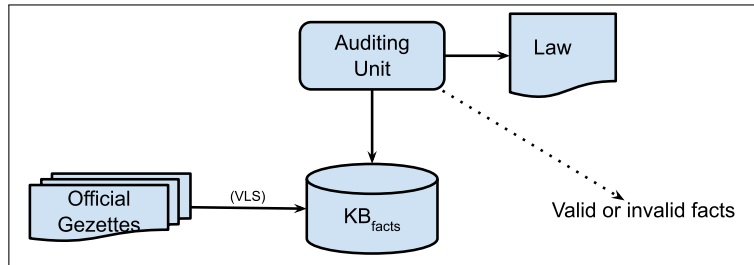


Figure 1.3: Overview of architecture propose.

Figure 1.3 shows the main object of this thesis. Basically, the audit unit infer and validate the facts (the VLSs) in accordance with the legislation (law).

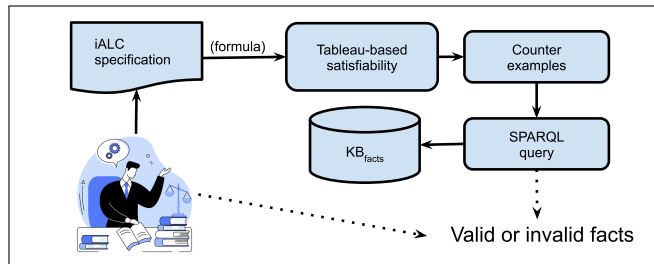


Figure 1.4: The execution flow of the inference process.

Figure 1.4 shows the functional flow of the audit process. An expert will formalize the audit functions in the formula *iALC*. Therefore, this formula will be available for a new tableau system developed for *iALC*. Then, based on the hypothetical counter-examples generated in the tableau inference process, SPARQL queries will be built to detect non-compliance in the real facts knowledge base.

This architecture can now extract information from well-structured government acts in the official gazette, such as the acts shown in Figures 1.1 and 1.2. In addition to this resource, we will propose a mechanism that uses attribute grammar to build basic formulas in *iALC* composed only of terminological concepts from this logic. And finally, we will present our inference system as an intuitionistic tableau for this logic. As stated before, its scope is reduced to inference in formulas composed only of terminological concepts.

## 1.1

### A framework vision

An important contribution of this thesis is the definition of a framework capable of facilitating the audit process in different knowledge bases from different domains. Its architecture can generalize communication between a logical theory and its class of models (databases), 1.5, providing a formal generic infrastructure for validation in information systems.

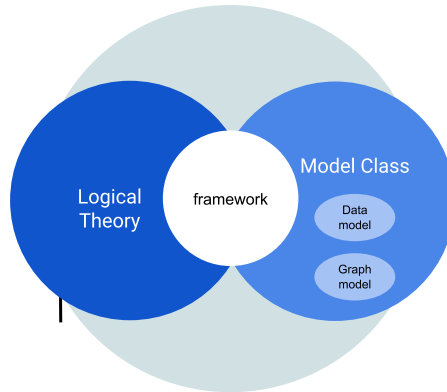


Figure 1.5: Class model framework.

Therefore, limiting the scope of the theory to just TBox results in a significant gain in computational complexity by reducing the range of problems to be solved to those that are verifiable in polynomial time. Focusing only on NP-Complete problems simplifies the analysis and solution of problems, ensuring that they are tractable in terms of execution time, even if they may be difficult to solve. This restriction allows for more efficient use of computational

resources, as it avoids the need to consider problems that require polynomial space, which can be more expensive in terms of memory usage.

As a Two-tiered KB Framework, we consider a Knowledge Base with a Model representing Data, also named KB with Data, as a pair  $(A, Q)$  where  $A$  is a theory presentation, i.e., a set  $A$  of formulas of some logic  $L_T$ , representing the theory  $Cn_{L_T}(A)$ , and  $Q$  is a model in  $L_M$ , where  $L_M$  is the logic where the models come from.  $Q$  is a model of  $A$ , i.e.  $Q \in Mod(A)$ .  $Cn(A)$  is the usual logical consequence operator.

We do not need to formally specify  $Cn$  since we consider the mappings  $Mod$  and  $Th$ . To be more flexible, we might consider a mapping  $\phi$  from non-logical languages in  $L_M$  to non-logical languages in  $L_T$  and the condition  $\phi_\Sigma(Q) \in Mod_\Sigma(A)$ .

We also need a validation procedure  $P_T$  that, whenever we give a  $\Sigma$ -formula<sup>1</sup>  $\alpha$  from  $L_T$ ,  $P_T$  verifies whether  $A \in Cn(\Sigma)(A)$  or not, providing the counter-examples belonging to the set  $CE_T$ , if  $A \notin Cn(\Sigma)(A)$ .  $CE_T$  is the set of all counter-examples to formulas in  $L_T$ .

To complete the formalization, we need a mapping  $q_\Sigma$  from  $CE_T$  into queries (formulas with free variables in  $L_M$ ) over  $Q$ , such that, for each  $\gamma \in CE_T(A, \alpha)$ ,  $q_\Sigma(\gamma) \not\subseteq Q$ . The framework is then formed by  $\langle (A, Q), P_T, q_\Sigma \rangle$ . We formally describe in the Section 1.2.

## 1.2

### The Formal Definition of the Framework

We might use a general or abstract concept of logic or logical systems. This research subject starts with *institutions* introduced by in [8] to define the notion of logical systems and other concepts such as *abstract logic*, as defined by [9]. It strongly uses Category Theory. Due to this and to better clarify our framework, we will consider the equivalent definition of the S-entailment<sup>2</sup> frame, defined below in a distilled version. We refer to [10], theorem 4.6, to verify that this notion is equivalent to other general and abstract logic notions in the literature.

**Definition 1.1** (S-Entailment Frame). *An S-ENTAILMENT FRAME  $\mathcal{L}$  is the 5-tuple  $(Sign, \mathcal{T}, \mathcal{M}, Mod, Th)$ , such that  $Sign$  is the category of signatures or non-logical languages,  $\mathcal{T}$  is the pre-ordered set of theory presentations, ordered by set inclusion,  $\mathcal{M}$  is the pre-ordered set of  $\mathcal{L}$ -models, ordered by the sub-model relationship, such that,  $Mod : \mathcal{T} \rightarrow \mathcal{M}$  and  $Th : \mathcal{M} \rightarrow \mathcal{T}$  are anti-monotone functions, for each  $i \in Sign$ , satisfying:*

<sup>1</sup>Formulas written in the non-logical language of  $L_T$ .

<sup>2</sup>The prefix S comes from the Set Theoretical Entailment Frame.



If  $\phi : \Sigma_1 \longrightarrow \Sigma_2$  is a map between signatures, then the following diagram commutes.

$$\begin{array}{ccccc}
 \Sigma_1 & & \mathcal{M}(\Sigma_1) & \xleftarrow{\text{Mod}(\Sigma_1)} & \mathcal{T}(\Sigma_1) \\
 \downarrow \phi & & \downarrow \mathcal{M}(\phi) & \xrightarrow{\text{Th}(\Sigma_1)} & \downarrow \mathcal{T}(\phi) \\
 \Sigma_2 & & \mathcal{M}(\Sigma_2) & \xleftarrow{\text{Mod}(\Sigma_2)} & \mathcal{T}(\Sigma_2) \\
 & & & \xrightarrow{\text{Th}(\Sigma_2)} & 
 \end{array}$$

The logical consequence operator is defined as  $Cn(\Sigma)(\Delta) = \mathcal{T}(\Sigma)(\mathcal{M}(\Sigma)(\Delta))$ . A  $\Sigma$ -formula is any formula written in a non-logical language  $\Sigma \in \text{Sign}_{L_T}$ .

We consider a pair of logics, or entailment frames as above,  $L_T$  and  $L_M$  used to express the theoretical and the model-based parts of the KB with data-model, and a logical mapping  $\phi$  from  $L_M$  to  $L_T$ , between S-entailment frames<sup>3</sup>, to link  $L_T$  theories and  $L_M$  models. Any pair  $(A, Q)$ , where  $A \in \mathcal{T}(\Sigma)$  and  $Q \in \mathcal{M}(\Sigma)$  that satisfies property 1.2 for some  $\Sigma \in \text{Sign}_{L_T}$  is called a KB with model data based on  $L_T$  and  $L_M$ .

Given an  $(A, Q)$  KB with model data, a validation procedure  $P_T$  is an algorithm that, if we give a  $\Sigma$ -formula  $\alpha$  from  $L_T$ ,  $P_T$  verifies whether  $A \in Cn(\Sigma)(A)$  or not, providing some counter-examples belonging to the set  $CE_T$ , the set of all counter-examples to formulas in  $L_T$ . To complete the picture, we have a mapping  $q_\Sigma$  from  $CE_T$  into queries (formulas with free variables in  $L_M$ ) over  $Q$ , such that, for each  $\gamma \in CE_T(A, \alpha)$ ,  $q_\Sigma(\gamma) \not\subseteq Q$ .

Given  $L_T$  and  $L_M$ , S-entailment frames, and a logical map  $\phi$  from  $L_M$  to  $L_T$ , we say that the pair  $(A, Q)$  satisfies the coupling Theory-Data in a signature  $\Sigma \in \text{Sign}_{L_T}$ , iff,  $\phi_\Sigma(Q) \in \text{Mod}_\Sigma(A)$  holds.

The framework for KD with data models, i.e., KB with DM, is then formed by  $\langle (A, Q), P_T, q_\Sigma \rangle$ , for some  $\Sigma \in \text{Sign}_{L_T}$ .

The first result within our framework for KB with DM is the theorem below that ensures whenever we have that property 1.2 holds for every signature  $\Sigma$  then the underlying function of the algorithm  $P_T$  is correct, what remains to have a framework is the effectiveness of this function. However, this is not always the case since  $L_T$  may not be decidable.

**Theorem 1.1.** *Given a class of frameworks,  $\langle (A, Q)_\Sigma, (P_T)_\Sigma, q_\Sigma \rangle_{\Sigma \in \text{Sign}}$ , such that, for each  $\Sigma \in \text{Sign}$ ,  $\langle (A, Q)_\Sigma, (P_T)_\Sigma, q_\Sigma \rangle$  satisfies property 1.2, then for*

<sup>3</sup>The S-specialized form of mapping of entailment frames, see [10].

each  $\Sigma \in \text{Sign}$  the underlying function of  $(P_T)_\Sigma$ , is correct, i.e., the mapping  $q_\Sigma$  maps counter-examples in  $L_T$  to counter-examples in  $L_M$ , via queries.

The whole toolbox defined in [10] can be used to detail<sup>4</sup> the KB theory with data models. However, this is out of the scope of this article, which aims to present these concepts only to show the relevance of their instantiations.

### 1.3 Computational Complexity

Another important contribution is the decrease in the computational complexity. Ontologies and OWL-based Semantic Web Technologies use TBox to represent the theoretical assertions and formalization of the domain-specific ontology. The model-based assertions are formalized using nominals in the ABox. We know that the computational complexity of TBox reasoning in the acyclic case is PSPACE-complete and, with the general case, is EXPTIME-complete and hence intractable, see [11] for the details cited here. There is a jungle of fragments of Description Logic with different complexities. Anyway, when considering only the validation in the TBox, i.e., with theoretical assertions only, the complexity is smaller than the overall complexity of TBOX+ABOX, for obvious reasons<sup>5</sup>. So, as is almost always the case in the practical validation of ontologies, this union-only way of integrating model-based and theoretical assertions is not very efficient. Since 2013, the usual way of dealing with ABox reasoning has been quite similar to FOL; the TBox is instantiated with the elements in the ABox, producing a greater TBox, and the validation goes on this new TBox. It is pretty much like checking satisfiability in first-order logic.

### 1.4 Abstraction of nonconformity detection

In context of inferring logical formulas and analyzing compliance with norms, it is essential not only to infer conclusions from these formulas but also to identify counter-examples on the database of facts that contradict these conclusions. Mapping these counter-examples into SPARQL queries by this framework is a crucial step for the analysis and validation of represented knowledge. Formally, this mapping can be defined by the function,

$$\text{Mapping} : C \rightarrow Q$$

<sup>4</sup>This is the aim of a forthcoming article.

<sup>5</sup>The size of the TBox+ABOX is strictly higher than the TBox only

where  $Mapping(C)$  is a function that maps the counter-model resulting from the inference process  $C$  into SPARQL queries  $Q$ .

Finally, the framework is capable of extracting real cases of nonconformity in the fact database. Formally, this audit mechanism can be defined as the function,

$$Auditing : Q \times F \rightarrow R$$

where  $Auditing(Q, F)$  is a function that, given the SPARQL queries  $Q$  and the knowledge base of facts  $F$ , extracts information  $R$  that contradicts the rules defined by the logical formulas.

This mapping to SPARQL, which can extract inconsistencies in an RDF fact base, is one of the basic features of this framework. By implementing this process, from a business audit perspective, we can detect and try to correct inconsistencies, standards violations, or compliance issues in systems based on formalized knowledge, thus contributing to more effective governance and informed decision-making.

Thus, this proposed framework has potential applications, including:

- General application for other data models and considering other theories and knowledge bases that could allow other interesting studies inferences.
- Specific verification of legal compliance in corporate governance systems.
- Specific detection of security breaches in information systems.
- Risk analysis in industrial and operational processes.

## 1.5

### Previous Results of Architecture

During the research, four articles were produced in trying to explain our approach. Firstly, in [12], we introduced the extraction information on the official gazette with regular expression patterns. The second paper [13] presented a combined approach with regular expressions and PEGs. In the third paper [14], we extended the scope to the auditing process on other 5 cities, and finally, the fourth article [15] showed a benchmarking of public information extractions between Machine Learning and Regular Language. In this work, we compare machine learning (ML) and rule-based approaches to the task of recognizing legal entities in the official gazette. We built an annotated dataset with 100 examples of legal documents and submitted this model to an evaluation in IBM Watson Knowledge Studio (WKS) [16]. In a scenario where

documents follow a formal structure, we show that rules-based information extraction systems still present themselves as low-cost, more uncomplicated, and more efficient solutions.

## 1.6

### Organization of Chapters

This document is organized as follows. Chapter 1, this introduction presents our research motivation as well as the challenges of developing this architecture. Chapter 2 shows the architecture for reasoning on legal norms, detailing the functionality of the components and their interactions. Chapter 3 shows the Legal Extraction and Normalization Module (ENM), a component capable of reading VLSs in the Official Gazette and serializing them into RDF triples. Chapter 4 shows the Formula Generator Module (FGM), a component capable of recognizing the constituent terms of legal norms and generating *iALC* formulas. Chapter 5, the major contribution of theses. We show our tableau system of *iALC* how engine for reasoning on legal norms. Still in this chapter, we show the solidity properties as proof of the quality of our SAT solver for *iALC*. Chapter 6, the evaluation and results, we carried out an experiment on the KRM module to evaluate two metrics: accuracy and query coverage. To achieve this, we built a set of competency questions with legally based *iALC* formulas. We discussed the results. Finally, Chapter 7 summarizes our contribution and discusses future work. We also produced an appendix with a) competency questions; b) the grammar (ANTLR) of the *iALC language*; c) an algorithm that calculates the accuracy of the experiment; and d) the soundness proof of the algorithm that implements the calculi system of the *iALC* tableau. We can also see the detailed result of the experiment in the document available in the [Experiment Report link](https://drive.google.com/file/d/1JdjhgctuAcuaq8EdS3wPibz-V1rMzcjX/view?usp=drive_link)<sup>6</sup>. In this report, there are for each competency question a) the proof tree images; b) the model or counter-example; c) the reference proof tree (proof tree developed manually); d) the SPARQL query generated by the tool. e) the RDF used in the experiment; f) the result set of executing the query in the RDF database.

<sup>6</sup>[https://drive.google.com/file/d/1JdjhgctuAcuaq8EdS3wPibz-V1rMzcjX/view?usp=drive\\_link](https://drive.google.com/file/d/1JdjhgctuAcuaq8EdS3wPibz-V1rMzcjX/view?usp=drive_link)

## 2

## Architectural Design for Reasoning

Currently, given advances in information management technologies, auditing systems are representatives in the most diverse business domains. A simple search on the Internet, for terms such as “auditing” or even “auditing systems”, shows us how common it is to use these systems to evaluate processes, checking whether the performance of routine functions is in accordance with the objectives, policies institutions, legislation, norms and required standards.

There are several applications of formal logical theories to the representation of legal rules, where the emphasis is on the representation and the (legal) conclusions arise from this representation through a process of deduction. For example: application of legal rules in unforeseen scenarios, interpretative reasoning according to the facts of a legal case, evidentiary reasoning to establish the facts of a case, formal models of legal procedures and multi-agent interaction in legal proceedings [17].

In this chapter, we will show the proposed architectural component system for our public data knowledge base auditing tool. The purpose of a system architecture is to create a conceptual structure that provides an overview of the system’s functionalities before the construction process begins. This is a crucial element in software development, in which a model is established, defining the system components and their interactions [18, p. 253].

In this sense, this architectural project is the first step in the process of developing our tool, being the link between the project and the technical specifications that identify the main structural components. Therefore, this architectural project aims to present how the main processing units (components) are organized and their integrations.

### 2.1

#### Related Works

As presented in the previous chapter, there are different approaches used in audit systems. In this section, we will emphasize works involving automatic reasoning using knowledge representation language for business process audit tasks. Thus, we use the phrases “**architectures for Knowledge Represen-**

tation and Reasoning (KRR)<sup>1</sup> on legal norms”, “Legal Knowledge Based Systems” and “Legal Knowledge Information Systems” as keys in the search process for related works. According to [20], the idea of knowledge representation, in the context of artificial intelligence (AI), is the ability to write representations of a world, in such a way that a machine can reach new conclusions about this world, manipulating these symbolic representations. Therefore, in the next works presented, we will find the application of some technique that uses some logical language as an inference mechanism (reasoning) on these knowledge representations.

We were unable to obtain more details about these tools because they are commercial industry products. Consequently, their investigation became restricted.

**ClauseMatch:** This system is capable of automatically comparing clauses and provisions in legal documents (contracts), identifying discrepancies and ensuring that they comply with current legislation. As it is a private property system, we did not have access to the theories applied to its solution.

**Compliance.ai:** Compliance.ai is a platform that uses artificial intelligence to monitor and analyze regulatory changes. Among some functions, this tool maintains a database of regulations, laws and standards, to facilitate research and cross-referencing. It also has a mechanism that generates compliance reports. As it is a private property system, we did not have access to the theories applied to its solution.

**LogicGate:** This is a risk and compliance management platform that uses automation and workflows to ensure compliance with regulations (legal standards) and internal policies (corporate rules). As it is a private property system, we did not have access to the theories applied to its solution.

During the research, we found other tools with purposes similar to those presented. But, as they are also private, we are unable to obtain details of their operation. We prefer not to present them.

<sup>1</sup>KRR is the area of Artificial Intelligence concerned with how knowledge can be represented symbolically and manipulated in an automated way by reasoning programs. [19]

## 2.2

### Architectural Design

This project has been developed over the last 3 years. During this time, we used an approach where we divided the main problem into small sub-problems. Each of these had its own time for research development and testing. The result was the implementation of a modular solution.

Basically, there are 3 cores that try to cover functionality in our approach, as shown in figure 2.1.

- Legal Extraction and Normalization Module (ENM);
- Formula Generator Module (FGM);
- Knowledge Base Reasoner Module (KRM);

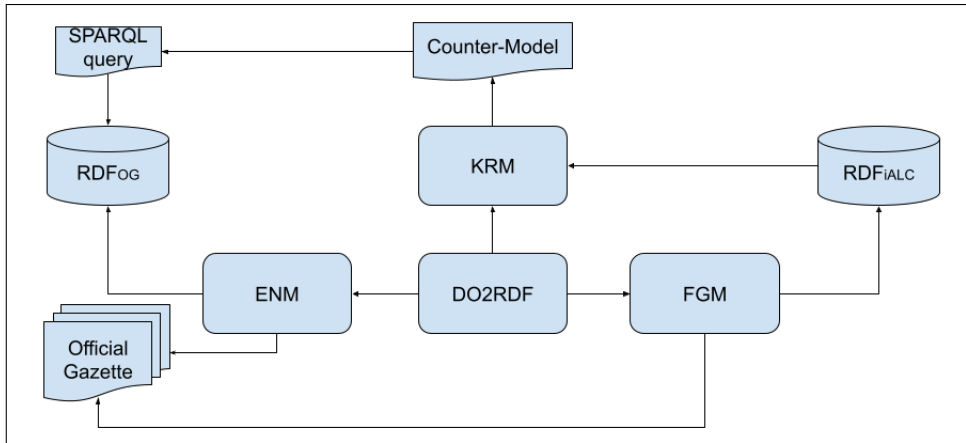


Figure 2.1: Architectural context diagram.

- **ENM** - It is responsible for automatically retrieving the Official Gazettes (OG) in PDF format and using regular expressions from its grammar to retrieve public acts. Then, they are serialized into RDF triples, composing a knowledge base for SPARQL queries. More details of this module will be presented in chapter 3.
- **FGM** - It is responsible for obtaining the entities extracted in the ENM module and converting them into *iALC* formulas. Specifically, given the complex nature of formalizing texts in natural language, we are unable to obtain formulas that express the intention (Objective) of what we want to audit. Therefore, it is necessary to formalize it in a semi-automatic (expert) way. These features, as well as the details of this module, will be presented in chapter 4.

- **KRM** - It is the main architectural module. It is responsible for applying the rules (*iALC* formulas that model the norm) established by ENM and solving them through a SAT solver built for *iALC* logic. As a result, a model (counter-example models) is created for SPARQL queries in our RDF knowledge base. The chapter 5 will present in detail the logical theory involved in this module. Also, in this chapter, we will present the system of rules for the SAT solver and the formalism that transforms counter-examples of this solver to SPARQL queries. Furthermore, we will show some formal quality aspects of the SAT solver rules, as well as the details of their implementation.



### 3

## Legal Extraction and Normalization Module (ENM)

As seen in the previous chapter, our proposed architecture is basically composed of 3 large modules: The Legal Extraction and Normalization Module (ENM); Formula Generator Module (FGM) and Knowledge Base Reasoner Module (KRM).

In this chapter we will show ENM, a component of our architecture responsible for extracting, classifying and serializing data contained in Official Gazettes, for an RDF knowledge base.

During our research, we observed that the official gazette texts could be classified as semi-structured data. Some sections of the text present a regularity in the disposition of the data, while others appear in a poorly structured way. For example, legal statements that distribute public office are published in well-formatted (structured) sections. The more general information in the official gazette is treated in an unstructured way. In [21], the authors present an ML approach to segmentation in an attempt to minimize the complexity of identification and extraction in semi-structured documents. In section 3.5 we present our approach with regular expressions.

As the official gazette is a semi-structured document, we question whether the Regular Expression (RE) and Parsing Expression Grammar (PEG) approaches are viable alternatives when compared to an ML approach in the tasks of recognizing entities contained in the official gazette.

### 3.1

#### Well-Formatted Document

Our objective has always been motivated by the ability of these well-formatted documents, containing formatted legal texts, to reflect a formalism that allows us to apply techniques that map their grammar to rule-based extraction systems.

A clear example of this category of documents is the official gazettes. Generally, these documents do not have a formal organization of sections. Each official gazette, published daily, discloses the most diverse categories of public administration acts. However, even though these documents seem to lack any

formalism, some sections, such as those dealing with hiring people for dear audiences, follow some legal rules of publication as highlighted in Figure 3.1.

**O SECRETÁRIO CHEFE DA SECRETARIA MUNICIPAL DA CASA CIVIL**, no uso das atribuições que lhe são conferidas pela legislação em vigor,

**RESOLVE**

Exonerar, a pedido, **ANAMARIA CARVALHO SCHNEIDER**, matrícula 57/253.542-5, com validade a partir de 3 de fevereiro de 2014, do Cargo em Comissão de Coordenador I, símbolo DAS-09, código 039447, da Coordenadoria de Demandas Institucionais, da Subsecretaria de Gestão, da Secretaria Municipal de Saúde.

**RESOLUÇÃO "P" Nº 170 DE 13 DE FEVEREIRO DE 2014**

**O SECRETÁRIO CHEFE DA SECRETARIA MUNICIPAL DA CASA CIVIL**, no uso das atribuições que lhe são conferidas pela legislação em vigor,

**RESOLVE**

Exonerar **STAEEL CHRISTIAN RIANI FREIRE**, matrícula 60/293.276-2, do Cargo em Comissão de Gerente II, símbolo DAS-07, código 039394, da Gerência de Atendimento a Demandas, da Coordenadoria de Administração de Contratos de Gestão com Organizações Sociais, da Subsecretaria de Gestão, da Secretaria Municipal de Saúde.

**RESOLUÇÃO "P" Nº 171 DE 13 DE FEVEREIRO DE 2014**

**O SECRETÁRIO CHEFE DA SECRETARIA MUNICIPAL DA CASA CIVIL**, no uso das atribuições que lhe são conferidas pela legislação em vigor,

**RESOLVE**

Exonerar, a pedido, **ANAMARIA CARVALHO SCHNEIDER**, matrícula 57/253.542-5, com validade a partir de 3 de fevereiro de 2014, do Cargo em Comissão de Coordenador I, símbolo DAS-09, código 039447, da Coordenadoria de Demandas Institucionais, da Subsecretaria de Gestão, da Secretaria Municipal de Saúde.

**RESOLUÇÃO "P" Nº 170 DE 13 DE FEVEREIRO DE 2014**

**O SECRETÁRIO CHEFE DA SECRETARIA MUNICIPAL DA CASA CIVIL**, no uso das atribuições que lhe são conferidas pela legislação em vigor,

**RESOLVE**

Exonerar **STAEEL CHRISTIAN RIANI FREIRE**, matrícula 60/293.276-2, do Cargo em Comissão de Gerente II, símbolo DAS-07, código 039394, da Gerência de Atendimento a Demandas, da Coordenadoria de Administração de Contratos de Gestão com Organizações Sociais, da Subsecretaria de Gestão, da Secretaria Municipal de Saúde.

**RESOLUÇÃO "P" Nº 171 DE 13 DE FEVEREIRO DE 2014**

**O SECRETÁRIO CHEFE DA SECRETARIA MUNICIPAL DA CASA CIVIL**, no uso das atribuições que lhe são conferidas pela legislação em vigor,

**RESOLVE**

Exonerar, a pedido, **ANAMARIA CARVALHO SCHNEIDER**, matrícula 57/253.542-5, com validade a partir de 3 de fevereiro de 2014, do Cargo em Comissão de Coordenador I, símbolo DAS-09, código 039447, da Coordenadoria de Demandas Institucionais, da Subsecretaria de Gestão, da Secretaria Municipal de Saúde.

**RESOLUÇÃO "P" Nº 172 DE 13 DE FEVEREIRO DE 2014**

**O SECRETÁRIO CHEFE DA SECRETARIA MUNICIPAL DA CASA CIVIL**, no uso das atribuições que lhe são conferidas pela legislação em vigor,

**RESOLVE**

Exonerar, a pedido, **ANAMARIA CARVALHO SCHNEIDER**, matrícula 57/253.542-5, com validade a partir de 3 de fevereiro de 2014, do Cargo em Comissão de Coordenador I, símbolo DAS-09, código 039447, da Coordenadoria de Demandas Institucionais, da Subsecretaria de Gestão, da Secretaria Municipal de Saúde.

**SECRETARIA DA CASA CIVIL**

Secretário: Pedro Paulo Carvalho Teixeira  
Rua Ademar Cavalcanti, 455 - 1ª andar - Tel.: 2976-3187

**DIRETORIA DE ADMINISTRAÇÃO E FINANÇAS**  
**DESPACHOS DO DIRETOR**  
**ERRATA**

Publicação do dia 10/02/2014 Pag. 6  
**01/800.126/2012**  
**Onde se lê:**  
6. Valor da despesa: R\$ 330.000,00 (trezentos e trinta reais)  
**Leia-se**  
6. Valor da despesa: R\$ 330.000,00 (trezentos e trinta mil reais)

**IPLANRIO**

Empresa Municipal de Informática S/A  
Av. Presidente Vargas, 3.131 - 12º andar - Tel.: 3971-1818 Fax: 3971-1589  
E-mail: iplanrio@pep-ij.gov.br

**DESPACHOS DO PRESIDENTE**  
**EXPEDIENTE DE 13/02/2014**

**PROCESSO Nº: 01/300.048/2014**  
**Objeto:** Ressarcimento de Despesa de Pessoal Requirido ao SERPRO.  
**Partes:** IPLANRIO e SERVIÇO FEDERAL DE PROCESSAMENTO DE DADOS - SERPRO  
**Fundamento:** Não sujeito a Lei 8666/93  
**Razão:** Não sujeito  
**Valor:** R\$ 1.049.000,00 (Um milhão e quarenta e nove mil reais).  
**Autorização:** VICTOR ZAJDAHAT, matr. Nº 45/620.889-7

**EXPEDIENTE DE 13.02.2014**  
Com base na manifestação do Órgão Gerenciador do Sistema de Registro de Preços da Iplanrio, autorizo os órgãos abaixo a fazerem uso dos preços registrados na Ata de Registro de Preços Nº 004/2013, conforme disposto no Decreto Municipal nº 36.567, de 04 de dezembro de 2012.

FSS 006/2013 e FSS 022/2013 - Secretaria Municipal de Desenvolvimento Social - SMDS  
Item 03 - 5.908 unidades Item 04 - 07 unidades Item 14 - 200 unidades Item 18 - 108 unidades Item 31 - 1.028 unidades Item 37 - 101 unidades

**Pensão**  
05/505.364/1994 - Marcelo Volpini Jansen Pereira  
Indefiro o pedido de Liberação de Reservas a fl. 158.  
05/500.471/1997 - Othoniel de Azevedo Blencourt  
Defiro o pedido de Extinção de Pensão a fl. 17.  
05/504.745/2004 - José Bastos  
Defiro o pedido de Extinção de Pensão a fl. 82.  
05/507.576/2013 - Adélia Fernandes  
Defiro o pedido de Reconsideração do Pagamento de Pensão às fls. 58 e 63.  
05/500.136/2014 - Itiel Natalino Moreira  
Defiro o pedido de Reconsideração do Pagamento de Pensão a fl. 26.  
05/507.820/2013 - Otton José Medeiros Brito  
Defiro o pedido de Pagamento de Pensão para Julia Lefebvre Brito Ferreira, a título precário a fl.19.  
05/507.301/2010 - Hamílcar Paschoa Silveira  
Indefiro o pedido de Extinção do Pagamento de Pensão a fl. 59.  
Defiro o pedido de Reversão do Pagamento de Pensão de Selma Ferreira de Andrade Silveira para Agatha Raissa de Andrade Silveira a fl. 60.

**Pecúlio Post Mortem**  
05/506.928/2013 - Robson Nascimento de Souza  
Defiro o pedido de Reconsideração do Pagamento de Pecúlio Post Mortem a fl. 19.  
05/500.424/2014 - Gilza Maria Silva Maia  
Defiro o pedido de Pagamento de Pecúlio a fl. 14.  
05/501.105/2014 - José Carlos Fajardo  
05/501.118/2014 - Lucine Nunes Ramada  
05/501.130/2014 - Francisco de Assis Araújo  
05/501.139/2014 - Maria Rosa da Silva Moura  
05/501.147/2014 - Diva Ferreira de Almeida  
Defiro o pedido de Pagamento de Pecúlio a fl. 02.

**Auxílio Funeral de Segurado**  
05/507.644/2013 - José Carlos de Souza  
Defiro o pedido de pagamento de Auxílio Funeral de Segurado a fl. 11.  
Indefiro o pedido de pagamento de Auxílio Funeral de Segurado a fl. 02.  
05/501.104/2014 - José Carlos Farjado  
05/501.107/2014 - Lucila Ferreira  
Defiro o pedido de pagamento de Auxílio Funeral de Segurado a fl. 02.  
05/500.519/2014 - Luiz de Oliveira  
Defiro o pedido de pagamento de Auxílio Funeral de Segurado a fl. 12

Ano XXVII • Nº 226 • Rio de Janeiro 4 Sexta-feira, 14 de Fevereiro de 2014

Figure 3.1: Example of a public act published in the Official Gazettes.

Figure 3.1 shows the kind of legal act available in the official gazettes. Therefore, these characteristics (structure) motivate us to question whether ML is a viable and efficient solution for such extraction tasks.

### 3.1.1

#### Syntactic structure of the law

In the context of the syntactic structure of Laws, legal norms are formally expressed in the form of propositions and may appear in documents of the legal system in the form of statements from which facts are verified. For example, according to [5, p. 81], the legal norm to which theft should be punished is often formulated by the legislator in the following proposition: “*Theft is punished with imprisonment;*”. On the other hand, a norm that grants the Head of State competence to conclude a treaty takes the form: “*The Head of State concludes an international treaty.*”.

Basically, a Brazilian normative text follows a well-formed struct publication, a template. This struct, which can be seen, looks like a syntactic structure. An attempt is made to standardize legal instruments in [3] and [4].

Thus, this normative gives rise to the structure:

#### 1. Preliminary Part

- (a) *Epigraph* - The title of the legal norm.
- (b) *Ementa* - The purpose of the legal norm (object).
- (c) *Preamble* - Institution for the practice of the act and its legal basis.

#### 2. Normative Part

- (a) *Substantive provisions*. Area intended for provisions pertaining to the measures necessary for the implementation of the rules (substantive content).

#### 3. Final Part

- (a) *Implementation* - Area intended for the provisions (dispositions) pertaining to the actions necessary for the implementation of substantive content legal norms.
- (b) *Transitory part* - Area intended for the transitory provisions (when they exist).
- (c) *Validity* - The laws have a period of validity that can be determined or undetermined. In its syntactic form, the laws that determine a period (“vacancy”) should use the wording [4]: “[...] *Esta lei entra em vigor após decorridos [the number of] dias de sua publicação oficial*”.
- (d) *Revocations* - when a law cites a “revocation”. It must expressly list the laws or legal provisions revoked (when exist) [4].

## 3.2

### Lexical and syntactic analysis

In this section, we address the importance of lexical and syntactic analysis in the context of regular grammar, highlighting the crucial role of regular expressions in this process. We begin by discussing the theoretical foundations behind lexical and syntactic analysis before exploring how regular grammar and regular expressions are applied for this purpose.

Lexical and syntactic analysis is a fundamental step in the process of understanding and processing formal languages. Our lexical analysis is focused on dividing the official gazette text into tokens that represent entities of a legal domain. For example, the registration number, the name of an official, and the date of a law. e.g. An example of these legal entities can be seen in Table 3.1. As for our syntactic analysis, we are concerned with the grammatical structure of publications in official journals in accordance with official rules. In practice, the combination of regular grammar and regular expressions is an essential activity to perform quick and efficient analysis of publications.

Here, it is interesting to remember that Regular Expression (RE) is a notation for specifying lexical patterns. Its syntactic construction is composed of atomic symbols (characters), union, concatenation, and Kleene closure of other regular expressions. Readers unfamiliar with the concept or terminology can refer to the book [22].

## 3.3

### Related Works

We found four articles discussing information retrieval in a similar context. The first three deal with the processing of official gazettes, and the last one shows a survey between industry and academia examining the choice of each technique. We discuss them below.

In the [23] uses NLP techniques, based on [24] to recognize Named Entities in appointment ordinance on Official Gazette. They use the resources available on the Natural Language Toolkit (NLTK) [25] platform for steps of the tokenization process until the entity recognition. A limitation of this work is that the authors present a tool that recognizes only the names of public agents (public employees) in appointment ordinances. In this experiment, it was possible to observe an accuracy of 92% in the extraction of names. Moreover, the details of which ML algorithms are used in this article are not reported in detail.

The [26] uses data mining techniques for information retrieval in the official gazette of the Government of Pernambuco, Brazil. This work reports

the application of the Random Tree algorithm with a hit rate of 80%. The authors agree that “if the department wants an algorithm with better results, it is necessary to carry out a minimum standardization of the Official Gazette so that the extraction is more efficient”. This highlights the complexity of the treatment of data contained in the official gazette. In this case, a study of other information retrieval strategies is necessary.

In [14], the authors use regular expressions to retrieve information that has been triplicated to an RDF format and can be further queried *SPARQL* against a triple storage database such as AllegroGraph [27]. Extracting acts from gazettes to a knowledge base is part of a broader project to create a KB for public documents in the context of electronic governance auditing and compliance.

The article [28] makes a case for the importance of the use of rules-based extraction systems for industry. It presents a research plan with the potential to bridge the gap between academic research and industry practice.

Finally, the article [15] shows a comparison of machine learning (ML) and rule-based approaches in the task of recognizing legal entities in the official gazette. In that work, the use of regular expressions was presented as a simple and efficient solution. It was observed that the ML preparation activities had a higher cost (time) than the preparation of rules in a regular expression. In that experiment, the process of extracting examples for annotation was a time-consuming activity. Even for a team carrying out this activity, the problem would be another issue: reliability and cost with workers.

### 3.4

#### Official Gazette Grammar

A central theory used for this module (ENM) was the mapping of the Official Gazette into a well-known formalism in the field of compilers. Parsing Expression Grammar (PEG) is a formalism that describes language recognizers and is a simpler alternative to presenting the syntactic formation rule (grammar) of certain languages. PEGs are stylistically similar to Context-Free Grammars (CFGs) [29] with RE-like features added, much like Extended Backus-Naur Form (EBNF) notation.

Here, we present an example of the PEG, Listing 1, identified in the Official Gazette. In this case, it seems reasonable that the Official Gazette becomes one of the primary sources of information for public administration, and its extraction becomes almost a necessity. Figure 3.2 shows examples of public acts that create a technical committee.

From the PEGs, it is possible to perform a direct translation into regular

**ATO DO SECRETÁRIO  
RESOLUÇÃO SMS Nº 2587 DE 14 DE ABRIL DE 2015**

Designa os membros da Comissão Técnica de Acompanhamento (CTA) do Contrato de Gestão nº 006/2011 referente ao processo instrutivo nº 09/000.008/2015 (Programa Saúde na Escola – PSE).

O SECRETÁRIO MUNICIPAL DE SAÚDE, no uso de suas atribuições que lhe são conferidas pela legislação em vigor.

CONSIDERANDO §2º do Artigo 8º da Lei Municipal nº 5026 de 19 de maio de 2009 que prevê a análise dos resultados atingidos com a execução do contrato de gestão por Comissão de Avaliação;

CONSIDERANDO a necessidade de monitoramento e avaliação das ações e serviços de saúde prestados pelas Organizações Sociais de Saúde que possuem contrato de gestão com a Secretaria Municipal de Saúde;

CONSIDERANDO a necessidade da revisão da composição da Comissão Técnica de Acompanhamento (CTA) do Contrato de Gestão.

**RESOLVE:**

**Art. 1º** Designar os membros abaixo indicados para comporem a Comissão Técnica de Acompanhamento:

COMPOSIÇÃO CTA – PSE		
TITULARES		
ÓRGÃO	NOME	MATRÍCULA
S/GAB	RODRIGO DO SOUSA PRADO	11/229.220-9
S/SUBPAV	LUCIANA SOARES RIBEIRO	11/227.277-1
S/ACS	CLAUDIA DE OLIVEIRA FARIA FERRARI QUADROS	11/157.497-9
S/SUBG	ANTONIO RICARDO GOMES JUNIOR	60/274.497-7
S/SUBG	MICHELLE RODRIGUES SCHINKE	11/226.675-7
SUPLENTE		
ÓRGÃO	NOME	MATRÍCULA
S/SUBPAV	NINA LUCIA PRATES NIELEBOCK	57/160.631-8
S/SUBG	ANA CAROLINA HENRIQUE SIQUEIRA LARA	60/262.710-7

**Art. 2º** Esta Resolução entra em vigor em 01 de agosto de 2014.

Rio de Janeiro, 14 de abril de 2015.

DANIEL SORANZ

Figure 3.2: Example of a public government act published in the Official Gazettes.

---

**Listing 1** Official Gazette grammar PEG.

---

```

⟨publicAct⟩  |=  ⟨top⟩⟨segment⟩
               |=  DECRETO "P" No.⟨port⟩ DE ⟨per⟩
               |=  ⟨per⟩ |=  ⟨day⟩ DE ⟨month⟩ DE ⟨year⟩
               |=  ⟨segment⟩ |=  ⟨segment1⟩⟨segment2⟩, símbolo⟨symbol⟩
               |=  ⟨segment1⟩ |=  RESOLVE ⟨act⟩⟨name⟩, matrícula⟨mat⟩,
               |=  ⟨segment2⟩ |=  ⟨compl⟩Cargo em Comissão de⟨publicfunc⟩
               |=  ⟨act⟩ |=  Nomear|Exonerar
               |=  ⟨name⟩ |=  [A-Z]+
               |=  ⟨port⟩ |=  [0-9]+
               |=  ⟨day⟩ |=  [0-9]+
               |=  ⟨month⟩ |=  [A-Z]+
               |=  ⟨year⟩ |=  [0-9]+
               |=  ⟨mat⟩ |=  [0-9/.-]+
               |=  ⟨compl⟩ |=  [A-Za-z0-9,-]+
               |=  ⟨publicfunc⟩ |=  [A-Z0-9]+
               |=  ⟨symbol⟩ |=  [A-Z0-9.-]+

```

---

expressions that represent the rules for extracting information. Due to the lack of space, we prefer to omit some development examples, but they can be easily seen in the project repository<sup>1</sup>. We believe that this formalism, present in the official gazette, makes this research quite reasonable.

**3.5****Official Gazette Patterns**

Table 3.1, shows the groups of regular expression where the act was “Nomear” (appoint) public servant in a job. We can observe the information to be retrieved: Ato (the public act), Nome (name), Matrícula (enrollment code), C. Efetivo (effective position), Dia (day), Mês (month), Ano (year), C. Comis. (commissioned position), Símbolo (symbol).

**3.6****ENM Modules**

Finally, Fig. 3.3 shows the high-level ENM module.

<sup>1</sup><https://github.com/fernandoantoniodantas/COMISSOES2RDF>

Entities	Patterns for Appointing Employee
Ato	RESOLVE[, \s]*Nomear\s+
Nome	(?P<nome>[A-ZÉÁÍÓÚÇÃÊÔÕÀÜ\s]+)
Matrícula	(?P<matricula>[0-9\./-]+)
C. Efetivo	(?P<cargoEfetivo>[A-ZÉÁÍÓÚÇÃÊÔÕÀÜa-zâêéóíçãâôú\-\s]+)
Dia	(?P<dia>[0-9]+)
Mês	(?P<mes>[J   j]aneiro   [F   f]evereiro   [...]   [D   d]ezembro)
Ano	(?P<ano>[0-9]+)
C. Comis.	(?P<cargo>[A-ZÉÁÍÓÚÇÃÊÔÕÀÜa-zâêéóíçãâôú\-\s]+)
Símbolo	(?P<simbolo>[A-Z\0-9\/\s]+)

Table 3.1: Regular expression pattern for the employee “Nomear” act.

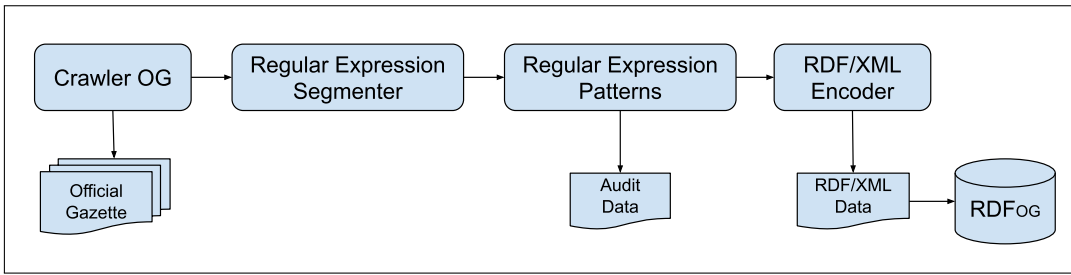


Figure 3.3: Integration among ENM components.

The modules are represented as follows:

- **Crawling Module** - It is responsible for retrieving the official gazette files automatically.
- **OG File** - Set of Official Gazette in PDF format.
- **Audit File** - File with the information extracted to facilitate the calibration process (accuracy) of the PEG production rules.
- **Regular Expression Patterns** - Regular expression patterns mapped from the official gazette PEG.
- **Regular Expression Segmenter** - Module for partitioning the Official Gazette to separate the target publications from the others. Providing the Regular Expression Patterns module with a segment with the detected grammar.
- **RDF/XML Encoding** - Module that transforms the extracted information into RDF/XML triples [30].
- **RDF/XML** - Serialization file with RDF/XML triples.
- **KB RDF** - Graph database for SPARQL queries containing RDF triples.



### 3.6.1

#### The Encoding Process

With the information extracted and matching it with regular expression patterns, we started the process of triplifying the information to an RDF format. The RDFLib [31] library for Python was used for this step. The library has interfaces that simplify and facilitate the implementation of RDF nodes. Optionally includes parsers for RDF/XML, N3, NTriples, N-Quads, Turtle, TriX, RDFa, and Microdata. It implements a Graph interface in which we can store graph information in memory or persistent storage. It is also possible to run queries and updates in the SPARQL language. In this work, we chose to serialize the data in RDF/XML format.

An essential step in the project was the ontology to give meaning to the contents of the Official Gazette. In such a case, as a basic proof of concept, we chose to define an adapted and generic ontology (Listing 2) in “*Friend of a Friend*” (FOAF) [32]. This ontology enabled queries in our knowledge base stored in *AllegroGraph* [33]. This Listing 2 shows the serialization process that translates data objects into RDF type.

---

#### Listing 2 Serialization process in *RDF/XML*

---

```
for row in comissoes_records:
    seqa+=1
    idP = seqa
    idP = BNode()
    store.add((idP, RDF.type, FOAF.Function))
    store.add((idP, FOAF.acao, Literal(row[0].strip())))
    store.add((idP, FOAF.dataPublic, Literal(row[1])))
    store.add((idP, FOAF.tipoCargo, Literal(row[2])))
    store.add((idP, FOAF.simbolo, Literal(row[3].strip())))
    store.add((idP, FOAF.dataEfeito, Literal(row[4])))
    store.add((idP, FOAF.matricula, Literal(row[5].strip())))
    store.add((idP, FOAF.cargo, Literal(row[6].strip())))
    store.add((idP, FOAF.nome, Literal(row[7].strip())))
    # Serialize the store as RDF/XML to the file DO2RDF.rdf.
    store.serialize("RDF/DO2RDF.rdf", format="pretty-xml", max_depth=3)
    print('RDF Serializations:', seqa, 'De', size)
```

---

For example, in the serialization process, we make the RDF triples file (DO2RDF.rdf) containing records of some government acts. The Listing 3 is a portion of this *DO2RDF.rdf* file, which shows the result serialization process with RDFLib. This file is used to deploy into AllegroGraph.

**Listing 3** Serialization in RDF/XML.

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:rdf="http://www.w3.org/1999/02/
22-rdf-syntax-ns#">
  <foaf:Function rdf:nodeID="N7654fa3584ea89c0ade">
    <foaf:acao>NOMEAR</foaf:acao>
    <foaf:dataPublic>2017-01-19</foaf:dataPublic>
    <foaf:tipoCargo>CC</foaf:tipoCargo>
    <foaf:simbolo>DAS-06</foaf:simbolo>
    <foaf:dataEfeito>2017-01-01</foaf:dataEfeito>
    <foaf:matricula>60/210917-1</foaf:matricula>
    <foaf:cargo>ASSISTENTE I</foaf:cargo>
    <foaf:nome>MARIA DOS SANTOS BASTOS</foaf:nome>
  </foaf:Function>
```

**3.6.2****The Audit File**

The result of implementing the RE was the extraction of information contained in 24,745 public acts. To facilitate the data analysis process, the tool generates an audit file (Fig.3.4) with the main information retrieved. This record, which presents data grouped by publication date, contains acts such as “Nomear” (recruit public employee), “Exonerar” (dismiss public employee). For example, as highlighted in the image, there was the publication of two acts for the public employee Antonio Barbosa: a dismissal from the commissioned position of Advisor II (DAS-08) and his recruitment to the commissioned position of Advisor I (DAS-09). Both on 01 Nov. 2013.

1 (PUC-RIO/TECMF)::PROCESSAMENTO DO DIÁRIO:: ANO: 26 No.:000201 TIPO:NORMAL \* RIO DE JANEIRO \* ARQUIVO: 1974.PDF SEQ.: 0001 26/02/2021 21:16:22

3	RESOLUÇÃO	0418	11/01/2013	NOMEAR	ARETUSA	PAULA	01/01/2013	DIRETOR I	DAS-09	CC
4	RESOLUÇÃO	0419	11/01/2013	NOMEAR	LILIANE	CESAR	01/01/2013	ASSISTENTE TÉCNICO	DAS-07	CC
5	RESOLUÇÃO	0420	11/01/2013	NOMEAR	JUSSARA		01/01/2013	DIRETOR IV	DAS-06	CC
6	RESOLUÇÃO	0421	11/01/2013	EXONERAR	60/25	-6	ANTONIO	BARBOSA	02/01/2013	ASSESSOR II
7	RESOLUÇÃO	0422	11/01/2013	NOMEAR	60/25	-6	ANTONIO	BARBOSA	02/01/2013	ASSESSOR I
8	RESOLUÇÃO	0425	11/01/2013	NOMEAR	LUIZ	PIRES	30/12/2012	ASSISTENTE I	DAS-06	CC
9	RESOLUÇÃO	0426	11/01/2013	EXONERAR	PAULA	CARMO	01/01/2013	ASSISTENTE I	DAS-06	CC
10	RESOLUÇÃO	0427	11/01/2013	EXONERAR	PAULO	PINTO	02/01/2013	ASSISTENTE I	DAS-06	CC
11	RESOLUÇÃO	0428	11/01/2013	NOMEAR	RENATO	LAGRIMANTE	02/01/2013	ASSISTENTE I	DAS-06	CC
12	RESOLUÇÃO	0435	11/01/2013	NOMEAR	MARIA	SANSON	02/01/2013	ASSESSOR II	DAS-08	CC
13	RESOLUÇÃO	0440	11/01/2013	EXONERAR	ALEXANDRE	BAPTISTA	01/01/2013	ASSESSOR III	DAS-07	CC
14	RESOLUÇÃO	0441	11/01/2013	EXONERAR	ALEXANDRE	OLIVEIRA	01/01/2013	ASSISTENTE I	DAS-06	CC
15	RESOLUÇÃO	0443	11/01/2013	EXONERAR	HELIO	FURTADO	01/01/2013	COORDENADOR I	DAS-09	CC
16	RESOLUÇÃO	0445	11/01/2013	EXONERAR	SUEDENI	OLIVEIRA	01/01/2013	ASSESSOR II	DAS-08	CC
17	RESOLUÇÃO	0449	11/01/2013	NOMEAR	SIMONE	GUILHERMINO	01/01/2013	ASSESSOR III	DAS-07	CC
18	RESOLUÇÃO	0450	11/01/2013	NOMEAR	MARCIA	SILVA	01/01/2013	ASSESSOR III	DAS-07	CC
19	RESOLUÇÃO	0451	11/01/2013	NOMEAR	ZORAIDE	COSTA	01/01/2013	ASSISTENTE I	DAS-06	CC
20	RESOLUÇÃO	0361	07/01/2013	NOMEAR	ISABEL		01/01/2013	ASSESSOR CHEFE	DAS-08	CC
21	RESOLUÇÃO	0361	07/01/2013	NOMEAR	MARIA	SILVA	01/01/2013	ASSESSOR CHEFE	DAS-08	CC
22	RESOLUÇÕES	0095	08/08/2013	NOMEAR	VITOR	ALMEIDA	04/01/2013	DIRETOR IV	DAS-06	CC
23	RESOLUÇÕES	0096	08/08/2013	DESIGNAR	VANESSA	FERREIRA	XX/XX/XXXX	ASSISTENTE II	DAI-06	FG
24	RESOLUÇÕES	0100	08/08/2013	DESIGNAR	ANA	HORTA	XX/XX/XXXX	COORDENADOR PEDAGÓGICO	DAI-06	FG
25	RESOLUÇÕES	0101	08/08/2013	DESIGNAR	FERNANDA	SILVA	XX/XX/XXXX	DIRETOR-ADJUNTO	DAI-06	FG

EXONERAR	60/25	-6	ANTONIO	BARBOSA
NOMEAR	60/25	-6	ANTONIO	BARBOSA

Figure 3.4: Audit file fragment with information retrieved from the official gazette.

### 3.7

#### Evaluation and Results for ENM

As presented, this module extracts data and serializes it to create a knowledge base of public acts. For its proper evaluation, an experiment was designed focusing on SPARQL queries to this knowledge base. This way is interesting because, in future work, there will be integration of the module responsible for searching for cases of non-compliance (KRM) with standards stored in this KB, thus demonstrating its effectiveness.

#### 3.7.1

##### Scope

The first challenge was to capture official gazettes for a period. To make this module, the language Python, version 3, was used as a *backend* of the production tool and generation of the RDF triples of public acts published in the Official Gazettes.

In order to highlight the contributions, we will restrict ourselves here to the City of Rio de Janeiro<sup>2</sup>, Maceió<sup>3</sup>, Palmas<sup>4</sup>, Recife<sup>5</sup>, and Florianópolis<sup>6</sup>. The scope was restricted to publications involving appointments, dismissals, and appointments to commissioned positions.

As presented, the Python module was developed, as seen in Fig. 3.3 in order to automate the download of the official gazettes used in the project. In total, 1,126 official gazettes were recovered and treated. With these documents, the second step was to build the extractor of information contained in each official gazette. In this process, we used the *RE*<sup>7</sup> library from *Python* and a set of regular expression patterns to compose a set of rules based on the grammar (Listing 4) of acts published on special commissions. An example of this publication was seen in Figure 3.2.

To exemplify the use of our tool, we decided to treat a subset of human resources information, such as public jobs that do not require a contest for admission. Based on this scope, pattern extractors were developed using regular expression techniques applied to the grammar of the acts targeted in this research. Naturally, regular expression algorithms tend to be greedy and identifying the grammar of posts and mapping them to their corresponding regular expressions made this stage of the process very efficient.

<sup>2</sup><https://doweb.rio.rj.gov.br/>

<sup>3</sup><https://www.diariomunicipal.com.br/maceio/>

<sup>4</sup><http://diariooficial.palmas.to.gov.br/>

<sup>5</sup><https://dome.recife.pe.gov.br/dome/>

<sup>6</sup><https://www.pmf.sc.gov.br/governo/index.php?pagina=govdiariooficial>

<sup>7</sup><https://docs.python.org/3/library/re.html>

**Listing 4** Piece of Official Gazette grammar to special commissions act.

---

```

⟨publicAct⟩  =  ⟨top⟩⟨segment⟩
  ⟨top⟩      =  RESOLUÇÃO SMS No.⟨port⟩ DE ⟨per⟩
  ⟨per⟩      =  ⟨day⟩ DE ⟨month⟩ DE ⟨year⟩
  ⟨segment⟩  =  ⟨segment1⟩⟨segment2⟩
  ⟨segment1⟩ =  Designa os membros da ⟨cta⟩ do Contrato de Gestão nº⟨contrato⟩
  ⟨segment2⟩ =  referente ao processo instrutivo nº⟨numProcesso⟩⟨descContrato⟩
  ⟨cta⟩      =  Comissão Técnica de Acompanhamento⟨tpoComissao⟩
  ⟨port⟩     =  [0-9]+
  ⟨day⟩      =  [0-9]+
  ⟨month⟩    =  [A-Z]+
  ⟨year⟩     =  [0-9]+
  ⟨contrato⟩ =  [0-9/]+
  ⟨numProcesso⟩ =  [0-9/.]+
  ⟨descContrato⟩ =  [A-Z0-9/.-]+
  ⟨tpoComissao⟩ =  (CTA)

```

---

### 3.7.2 Results

We present in this section the results obtained after extracting the information contained in the gazettes and the triplifying step in *RDF/XML* data to be executed in *SPARQL* query language environment. Just to illustrate the use of our approach, we executed some *SPARQL* queries in this RDF graph environment.

The Listing 5, a simple *SPARQL* query that identifies the frequency of the same employee in the database. Our objective is to identify the employees with more than 1 participation in public committees.

**Listing 5** *SPARQL* queries for committee history.

---

```

SELECT (COUNT(?Matricula) as ?count) ?Matricula ?Nome
where{
  ?person foaf:name ?Nome .
  ?person foaf:matricula ?Matricula .

} group by ?Matricula ?Nome HAVING (?count > 1)
order by DESC(?count)

```

---

As shown in this Figure(3.5), the result of this *SPARQL* query retrieves some interesting cases for analysis. In the first line, employee Marcos [...] Santos participated in 151 committees. For more details about this case, the *SPARQL* query, Listing 23, retrieves some data: date of publication in the official

gazette, ordinance, process number, and the contract and its description for this employee, Figure 3.6.

count	Matricula	Nome
"151"	"11-6"	"MARCOS SANTOS"
"57"	"11-2"	"FERNANDO"
"39"	"11-0"	"RAPHAEL"
"37"	"11-2"	"CARLA"
"35"	"11-3"	"CONRADO"
"31"	"11-2"	"CRISTINA"
"25"	"11-5"	"HUGO"
"22"	"10-8"	"KATIA"
"20"	"11-4"	"ANA"
"19"	"10-7"	"IVY"
"19"	"11-0"	"NICOLE"
"18"	"11-9"	"CARLOS"
"18"	"11-1"	"MARIA"
"18"	"11-8"	"MARCIO"
"16"	"11-7"	"MICHELLE"

Figure 3.5: Number of participations in committees per employee.

---

**Listing 6** SPARQL query for a specific employee.

---

```

SELECT ?Data_Publica ?Portaria ?Numero_Processo
?Numero_Contrato ?Descricao_Contrato ?Matricula ?Nome
where{
?person foaf:nome ?Nome .
?person foaf:matricula ?Matricula .
?person foaf:dataPub ?Data_Publica .
?person foaf:portaria ?Portaria .
?person foaf:numContrato ?Numero_Contrato .
?person foaf:numProcesso ?Numero_Processo .
?person foaf:descContrato ?Descricao_Contrato .
FILTER (?Matricula='11/131.404-6') .
} order by ASC(?Data_Publica)

```

---

Data_Publica	Portaria	Numero_Processo	Numero_Contrato	Descricao_Contrato	Matricula	Nome
"2013-05-09"	"2067"	"09/004.994/09"	"005/2009"	"CAP-3.1"	"11-6"	"MARCOS SANTOS"
"2013-05-09"	"2066"	"09/004.993/09"	"006/2009"	"CAP-2.1"	"11-6"	"MARCOS SANTOS"
"2013-05-09"	"2065"	"09/002.080/11"	"008/2011"	"CAP-1"	"11-6"	"MARCOS SANTOS"
"2013-05-10"	"2071"	"09/005.019/09"	"004/2009"	"CAP-3.3"	"11-6"	"MARCOS SANTOS"
"2013-05-10"	"2070"	"0000"	"0000"	"0000"	"11-6"	"MARCOS SANTOS"
"2013-05-10"	"2074"	"0000"	"0000"	"0000"	"11-6"	"MARCOS SANTOS"
"2013-05-10"	"2075"	"0000"	"0000"	"0000"	"11-6"	"MARCOS SANTOS"
"2013-05-10"	"2073"	"0000"	"0000"	"0000"	"11-6"	"MARCOS SANTOS"
"2013-05-10"	"2072"	"09/004.435/10"	"004/2011"	"CAP-4"	"11-6"	"MARCOS SANTOS"
"2013-05-10"	"2069"	"0000"	"0000"	"0000"	"11-6"	"MARCOS SANTOS"
"2013-05-29"	"2084"	"09/004.435/10"	"004/2011"	"CAP-4"	"11-6"	"MARCOS SANTOS"
"2013-05-29"	"2085"	"09/004.436/10"	"002/2011"	"CAP-5.1"	"11-6"	"MARCOS SANTOS"
"2013-05-29"	"2087"	"09/004.603/09"	"001/2009"	"CAP-5.3"	"11-6"	"MARCOS SANTOS"
"2013-05-29"	"2086"	"09/004.437/10"	"003/2011"	"CAP-5.2"	"11-6"	"MARCOS SANTOS"
"2013-05-29"	"2082"	"09/004.991/09"	"020/2010"	"CAP-3.2"	"11-6"	"MARCOS SANTOS"

Figure 3.6: Piece of history of participation in committees.

Finally, our last query (Listing 7) aims to retrieve information relating to employees and their participation in committees, as well as the period in which they participated or participated in these committees.

The results of this query are shown in Figure 3.7, where the employee Marcos [...] Santos participated or participated for eight years in committees.

**Listing 7** SPARQL queries for time in the committees.

```

select (COUNT(?Matricula) as ?count) ?Matricula ?Nome
(min(?Data_Publica) AS ?min) (max(?Data_Publica) AS ?max)
(year(?max)-year(?min)AS ?anos)
where {
  ?person foaf:name ?Nome .
  ?person foaf:dataPub ?Data_Publica .
  ?person foaf:matricula ?Matricula .
} group by ?Matricula ?Nome ?idade HAVING (?count > 1)
order by DESC(?count)

```

count	Matricula	Nome	min	max	anos
"151"	"11.000.000-6"	"MARCOS FELIPE DOS SANTOS"	"2013-05-09"	"2021-08-10"	"8"
"57"	"11.000.000-2"	"FERNANDO ALVES DOS SANTOS"	"2015-06-12"	"2019-06-05"	"4"
"39"	"11.000.000-0"	"RAPHAEL ALVES DOS SANTOS"	"2014-09-26"	"2018-05-30"	"4"
"37"	"11.000.000-2"	"CARLA LOPES DOS SANTOS"	"2013-08-12"	"2020-09-17"	"7"
"35"	"11.000.000-3"	"CONRADO RODRIGUES WEBER JUNIOR"	"2014-09-26"	"2019-06-05"	"5"
"31"	"11.000.000-2"	"CRISTINA RODRIGUES DOS SANTOS"	"2015-12-30"	"2019-07-17"	"4"
"25"	"11.000.000-5"	"HUGO RODRIGUES DOS SANTOS"	"2014-09-26"	"2018-05-29"	"4"
"22"	"10.000.000-8"	"FABIANA DOS SANTOS SILVA"	"2014-09-30"	"2016-04-28"	"2"
"20"	"11.000.000-4"	"ANA RODRIGUES DOS SANTOS"	"2016-09-15"	"2019-05-24"	"3"
"19"	"10.000.000-7"	"MARCIA RODRIGUES"	"2015-06-12"	"2016-05-30"	"1"
"19"	"11.000.000-0"	"NICOLE RODRIGUES DOS SANTOS"	"2014-02-17"	"2018-11-28"	"4"
"18"	"11.000.000-1"	"MARIA CRISTINA RODRIGUES DOS SANTOS"	"2013-05-29"	"2019-01-14"	"6"
"18"	"11.000.000-8"	"MARCIO DE CARVALHO DOS SANTOS"	"2018-04-04"	"2018-06-15"	"0"
"18"	"11.000.000-9"	"SILVIA RODRIGUES DOS SANTOS CARLOS"	"2018-04-04"	"2018-06-15"	"0"
"16"	"11.000.000-7"	"MICHELLE RODRIGUES DOS SANTOS"	"2018-04-04"	"2018-05-29"	"0"

Figure 3.7: Piece of history of participation in committees per year.

We presented how extracting information from PDF documents can help in the process of continuous auditing of public acts available in the Official Gazettes. Due to the characteristics of the Official Gazette, the use of regular expressions was presented as a simple and efficient solution. One of the challenges of implementing was the use of the *RDFLib*. This lib showed little flexibility for defining new ontologies. Nevertheless, the solution was to adapt the FOAF ontologies to the characteristics of this research. Finally, the serialization in RDF/XML format became efficient for research purposes and *SPARQL* queries were performed to demonstrate the reasonableness of the tool.

### 3.8

#### An Alternative with Machine Learning

During this research, we question whether this technique is the best solution for extracting information from documents. We compare machine learning (ML) and rule-based approaches in the task of recognizing legal entities in the official gazette. For this, We built an annotated dataset with 100 examples of legal documents and submitted this model to an evaluation in IBM Watson Knowledge Studio (WKS). We show that, in a scenario where documents follow a formal structure, rules-based information extraction

systems still present themselves as low-cost, more uncomplicated, and more efficient solutions. The next section shows these results.

### 3.8.1 Results

An important aspect is the definition of the scope of this research. Traditionally, the industry makes its benchmarking of techniques based on these five criteria: Data Capacity, Training Speed, Model Precision, and Inference Speed. In this research, we adopted the **Model Precision** and a new task, the **Development Time** for both approaches (rules-based vs ML).

For this benchmarking, our experiment reproduced the same scenario and techniques presented in subsections 3.5 and 3.6.2. The only difference is that now we train a model (by Machine Learning) and apply it to the process of extracting named entities from the official gazette.

As a result, all the examples used in the training set of our NLP machine were recognized by our rule-based system. The machine learning model obtained an accuracy of 0.99. The Table 3.2 summarizes these results.

Approach	Examples	Accuracy
ML-based	100	0.99
Rule-based	100	1.00

Table 3.2: Result between the two techniques.

The time of preparation of the experiment was also analyzed. We only compare the phase that precedes the execution of the extraction algorithms. Therefore, we did not evaluate the machine processing time of both approaches. Four essential steps were observed: time to select examples, time to annotate the examples, develop rules, and review.

We can see in Table 3.3 that the Machine Learning-based approach for this task is more costly than the rule-based one. We emphasize that the preprocessing of the experiments was carried out by one person.

Approach	Task	Time
ML-Based	Selection of examples for annotation	360 min.
ML-Based	Annotation process	240 min.
ML-Based	Review	120 min.
Rules-Based	Rules development	240 min.
Rules-Based	Review	120 min.

Table 3.3: Preprocessing experiments time.

In addition, the black-box aspect of the process inherent to statistical approaches makes it necessary, many times, to have a new treatment of the training set or adjustments to the model's hyperparameters. We are leading to more costs during extraction preparation. In this project, 12 hours were consumed to prepare the training set. It took 6 hours to prepare the extraction rules. Table 3.3 presents the details of time measurement.

Based on the presented, we have to conclude that rule-based extraction techniques can easily replace this ML practice. In scenarios where documents have a well-defined grammar (some structure), rules-based information extraction systems still present themselves as low-cost, simpler, and more efficient solutions.

One of the contributions of this project was the annotation of these 100 examples (made by specialists). This dataset is available for use in other experimental research. Our idea is to add new examples, types, and relationships to this base. More details about this research can be found in [15].



## 4

### Formula Generator Module (FGM)

As seen in the previous chapter, the data extracted by ENM are entities that have some semantic meaning. In that extraction process, the entities contained in the official gazette are duly recognized and labeled. We start from the idea that the matching of syntactic patterns in the official gazette follows from some deterministic choice of a previously defined dataset of regular grammars.

This chapter is intended to show the component of our architecture that converts (semi-automatically) natural Portuguese law texts into *iALC* formulas via a series of structural representations. The central idea is to present a proposal for a mechanism that uses pattern recognition techniques (lexical and syntactic) from the grammar of legal norms. Basically, the FGM module uses the entities extracted in ENM module and converts them into *iALC* formulas. To do this, we use Attribute Grammar, a classic compiler technique used to generate new information during the compilation process.

The attribute grammar is a method applied in formal language theory to delineate the grammatical structure of an artificial language. It delves into the relationship between words and sentence structures, assigning properties and values to each element within the sentence.

The algorithm applied to attribute grammar defines the linguistic properties that can be assigned to a non-terminal symbol. These attributes encapsulate values expressing specific information about the symbol, calculated through rules establishing relationships between symbols and their attributes. Essentially, the aim is to confer meaning to strings in a language, representing this meaning as “attributes” to the symbols in an Abstract Syntax Tree (AST) of that string [34].

A distinguishing characteristic of attribute grammar lies in its capacity to analyze more intricate syntax, considering not only the sentence’s structure but also the properties and traits of its constituent elements. This enables a more precise and detailed depiction of linguistic structures. In parsing with attribute grammar, a set of rules needs to be established to delineate the properties and values attributed to each non-terminal symbol. These rules can be articulated in a formal language, such as attribute definition language.

This grammatical approach finds utility across various domains in linguistics and computer science [35], facilitating both the analysis and description of natural languages and the development of natural language processing systems. In [36] elucidates how this process unfolds concerning attributes that are “synthesized”, when defined solely in terms of attributes of the corresponding non-terminal symbol’s descendants and attributes that are “inherited”, when defined based on attributes of the non-terminal symbol’s ancestors.

During this research, a morphological approach was experimented with using machine learning. Coreference Resolution [37] is a technique that consists of identifying the different forms and relationships between named entities in a given text written in natural language. In practice, its algorithm identifies terms (entities) and expressions (via anaphors and cataphors) and tries to find the relationships between the entities, which are distributed in different parts of a text. Coreference Resolution serves as a vital component for various advanced NLP tasks, including document summarization, question answering, and Information Extraction (IE). The task of extracting data (entities) from the laws contained in the official gazette and finding their relationships to formalize them in a language of description proved to be, at first, quite promising.

We found a fundamental problem for its application in this research: the lack of a trained model, with laws, for the Portuguese language.

Building a golden dataset for coreference resolution can be a challenging and arduous task [38]. There are several reasons for this, as main examples: *Manual annotation requires team expertise* - Manually annotating coreference chains in a text requires linguistic knowledge and an understanding of the nuances of reference and textual cohesion. This often requires trained annotators or linguistic experts. *Review and quality assurance* - Ensuring the consistency and quality of annotations is essential to building a reliable reference dataset. This often requires careful review and multiple annotation iterations.

To minimize this effort, we tried to adapt a Portuguese coreference model (Corref-PT [39]) and Summ-it++ [40] to the format used in Conference on Computational Natural Language Learning (CoNLL) competitions. We use the NeuralCoref2<sup>1</sup> multi-language training suite. In [42] we can see an application of this technology for the Chinese. The result was not what was expected. We noticed in a visual inspection the low quality of the annotations in corref-pt, as well as the great difficulty in adapting the suite to the model training processes.

After this long journey in an attempt to create a coreference model for

<sup>1</sup>NeuralCoref is a pipeline extension for spaCy [41] which annotates and resolves coreference clusters using a neural network with multi-language extension options.

Laws, we chose to continue with an approach analyzing legal texts from a formal perspective. As presented in chapter 3, we can associate each law with a formal grammar that interprets it. Therefore, we defined that for each law an associated attribute grammar would be sufficient to generate *iALC* formulas at the time of their syntactic analysis by an *iALC* compiler. For practical reasons, this step is a theoretical presentation of the Formula Generation Module (FGM).

#### 4.0.1 FGM Modules

In this section, we present the main components of FGM Modules. The purpose of these integration components is to provide a support structure for the construction of formulas in *iALC*. To achieve this, we present the architecture at a high level by integrating its modules Fig. 4.1.

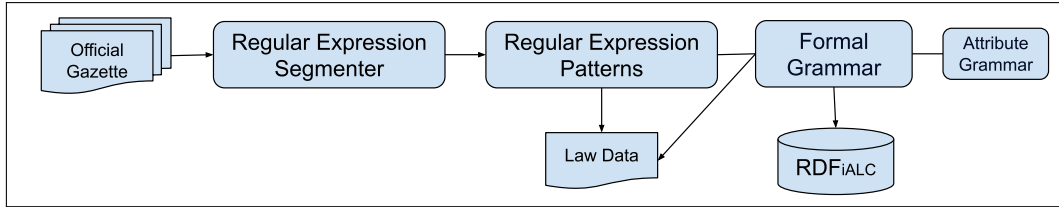


Figure 4.1: Integration among FGM components.

The modules are represented as follows:

- **Official Gazettes** - Set of Official Gazette files in PDF format.
- **Regular Expression Segmenter** - Module for partitioning the Official Gazette, separating the target publications from the other publications. Providing the Regular Expression Patterns module with a segment with the detected grammar.
- **Regular Expression Patterns** - Regular expression patterns mapped from the official gazette PEG for laws. Similar to the applied in ENM Module.
- **Formal Grammar** - Alphabet, terminal and non-terminal symbols, production rules that specify how a valid legal statement is formed.
- **Attribute Grammar** - An extension of formal grammar associating attributes with grammar productions. These attributes are computed values that can be assigned to non-terminal or terminal symbols in that grammar. We chose to perform the AST analysis and code generation (formula) using the synthesized attributes technique. This technique consists in assigning values to the attributes of the AST nodes from

their leaves toward the root. The next chapter will present a practical example of this technique.

- $RDF_{iALC}$  - Graphical database for containing RDF triples with formulas extracted from the laws. These formulas represent a formal description of the legal rule. Basically, they are our Knowledge Representation Base (KRB).

## 4.1

### From Law Text to ASTs

As shown, for each law, its grammar is associated with semantic rules to assist in the construction of formulas. In this subsection, we will present the basic component of this module, the formal grammar, and the semantic rules of a state law that regulates the taxation of taxes on products and services.

We start with the mapping by presenting in Figure 4.2 where on the left side are the Production Rules and, for each line there is an associated Semantic Rule (right side). Basically, each semantic rule has parts of formula code in the  $iALC$  language. In practice, when an AST analyzer starts its visit from the leaves, for each production rule used a part of the  $iALC$  formula is constructed. This process is done recursively up to the root of the AST.

Production Rules	Semantic Rules
$S \rightarrow NORMA$	$\{Imp_1.v := CM.v \sqcup PS_3.v\}$
$NORMA \rightarrow CM$	$\epsilon$
$NORMA \rightarrow PS$	$\epsilon$
$CM \rightarrow "mercad_1"$	$\{CM.v := mercad_1:CM_2\}$
$CM \rightarrow "mercad_2"$	$\{CM.v := mercad_2:CM_2\}$
$CM \rightarrow "mercad_n"$	$\{CM.v := mercad_n:CM_2\}$
$PS \rightarrow T COM$	$\{PS_3.v := T.v \sqcup Com_5.v\}$
$T \rightarrow Te Tm$	$\{T.v := TranspA:Te \sqcup TranspB:Tm\}$
$Te \rightarrow "TranspA"$	$\{Te.v := TranspA:Te\}$
$Tm \rightarrow "TranspB"$	$\{Tm.v := TranspB:Tm\}$
$Com \rightarrow "ComA"$	$\{Com.v := ComA:Com\}$
$STR \rightarrow [a - ZA - Z]^*$	$\epsilon$

Figure 4.2: Attribute Grammar Tables.

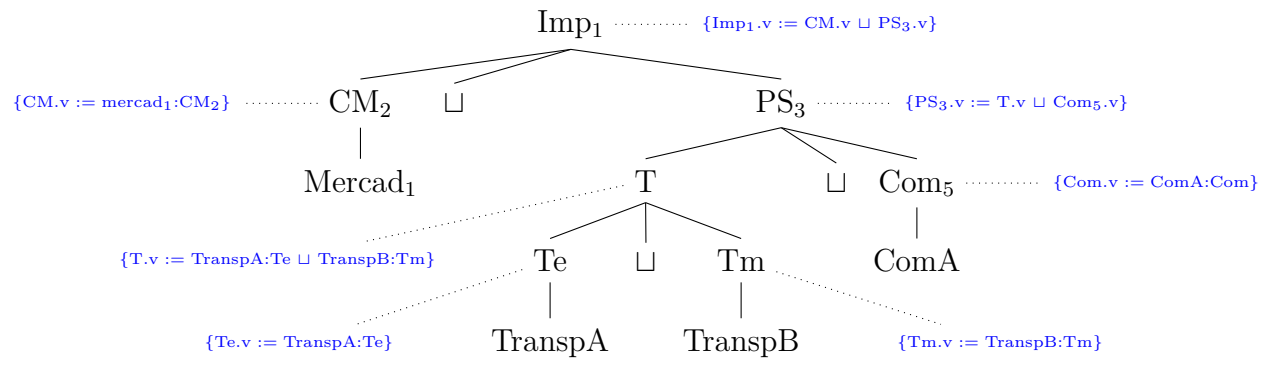


Figure 4.3: Abstract Syntax Tree (AST) of the attribute grammar of a state tax law.

The AST Figure 4.3 shows this process in detail. Note that from the leaf to the root the attributes are associated with generating the formula  $CM \sqcup PS$  (at the root), it is a conjunction between the *Concepts* “CM” (circulação de mercadorias) and “PS” (produtos e serviços) in the law.

This was just a presentation of a formula generator proposal. It presents how entities from a legal domain, in the context of a specific law, must be combined and transformed into basic *iALC* formulas. This module deserves more details on some practical points such as which binary logical symbol of *iALC* will be applied in the composition of the formulas. Note that they appear in the AST leafs, but we do not actually discuss how to extract such grammatical information from the legal text. A deep dive into the grammatical structures of law may be necessary to identify these logical connectives between legal entities. Furthermore, we are also not concerned with the *Roles* definition of *iALC* concepts. These topics are for future work.

## 5

### Knowledge Base Reasoner Module (KRM)

As seen in the previous chapter, the purpose of FGM module is to build formulas based on the grammar of the official gazette. The choice of grammar among a set of grammars depends on the degree of similarity between the entities extracted in the ENM module and the terminals of a regular grammar ( $g \in G$ ) in the FGM module.

In this chapter, we will show the component of our architecture that, given an *iALC* formula, which represents the audit objective, finds models (counter-example) for inference in a knowledge base containing public acts. Thereby maintaining some level of public compliance.

The Knowledge Base Reasoner Module (KRM) is an essential tool for performing logical inferences on a Knowledge Representation Base (KRB) written in *iALC* formulas. This module uses an *iALC* tableau calculi system to verify the validity of the rules present in this Knowledge Representation Base, seeking to identify possible counter-examples. Then, a parser transforms these counter-examples into SPARQL queries, aiming to detect nonconformities with the legal norms within the knowledge base of facts. This efficient and automated approach makes KRM an indispensable tool for ensuring consistency and compliance of information within a knowledge base. The KRM modules will be shown in the next section.

#### 5.1

##### KRM Modules

In this section, we will show the main elements that compound the KRM. To facilitate the understanding of how these components interact, we present Figure 5.1, an abstraction of the architecture of this module and its structural design. This module integrates 4 major components: auditing unit, query handling unit, compliance specifications, and facts unit. The last two are implemented by ENM and FGM modules.

- **The Auditing Unit** - This unit implements the inference engine for formulas written in *iALC*. The SAT solver is a tableau system that extends the first-order intuitionistic tableau presented by Fitting (1960) [43]. Basically, this solver searches for models and counter-models.

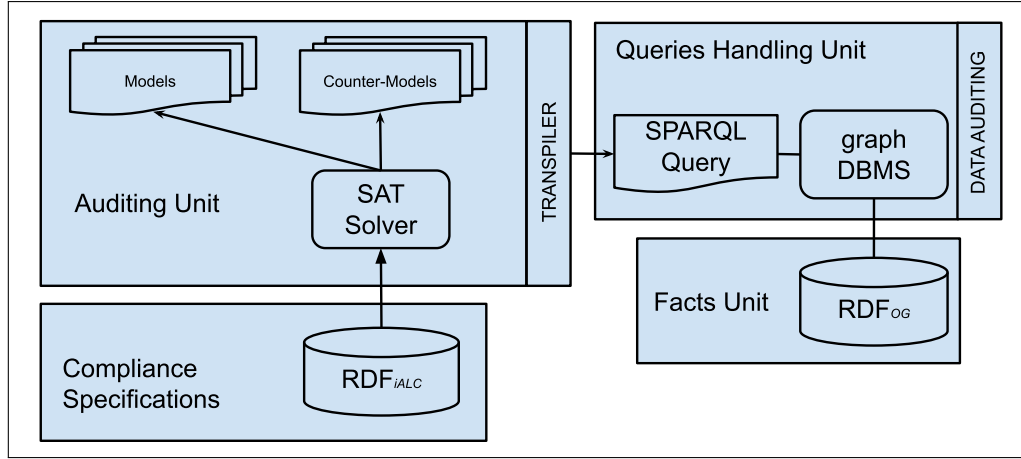


Figure 5.1: Integration among KRM components.

The last is the basis for generating preliminary query code in SPARQL. Details of its implementation will be presented in sections 5.4 to 5.6.

- **Queries Handling Unit** - Its operation involves receiving the translation generated by the transpiler, which converts the counter-model grammar extracted from SAT Solver to queries in SPARQL, a query language used in environments linked to ontologies and data semantics. Once these steps are completed, the module can then forward the resulting query to the semantic data query platform, ensuring harmonious integration between the different systems (Logical Rules to Graph Database) and allowing the extraction of significant information for the process audit. Details of its implementation will be presented in subsection 5.6.3.
- **Compliance Specifications** - Unit that stores the formulas (by FGM module) that will be input for reasoning in the *iALC* tableau. It is a Knowledge Representation Base (KRB).
- **Facts Unit** - Unit that contains the RDF triples of facts extracted from official gazettes (by ENM module).

## 5.2

### Logical-base components

Before presenting *iALC*, the logic behind our inference mechanism, we will present the main concepts of description logic.

**Description Logics (DL)** is a formal language used for knowledge representations and reasoning about it [11]. Essentially, DL has three components: *Individuals* (constants) that represent entities in a domain; *Concepts* (unary predicates) that are properties given to Individuals; *Roles* (binary predicates), which are the relationships between Individuals [44]. This language is based on



first-order logic and its objective is to provide a precise and semantic formal language to represents knowledge in a domain. For this, we show a piece of the alphabet of a DL consisting of:

- Set of names of Individuals, Concepts<sup>1</sup> and Roles.
- $\sqcap$  represents the conjunction of Concepts.
- $\sqcup$  represents the disjunction of Concepts. The operation  $C \sqcup D$  is used to restrict the set of individuals under consideration to those that belong to both  $C$  and  $D$ .
- $\sqsubseteq$  represents the inclusion of Concepts. The operation  $C \sqsubseteq D$  indicates that Concept  $C$  is included in Concept  $D$ .
- $\forall$  represents the universal restriction of Concepts. Thus,  $\forall R.C$  represents the universal restriction of the Concept  $C$  under Role  $R$ . This operation requires that all the individuals that are in the relationship  $R$  with the concept being described belong to the concept  $C$ . In short, all individuals who stand in relation  $R$  to an individual described by the concept in question are also describable as  $C_s$  [11].
- $\exists$  represents the existential restriction of Concepts using Roles. Similar to universal restriction,  $\exists R.C$  is an operation that requires that at least one individual that is in the relationship  $R$  with the concept being described belongs to the concept  $C$ .
- $\neg$  represents the complement of Concepts.
- $:$  an assertion operator of type  $a : C$  (Concept assertions) and  $(a, b) : R$  (Roles assertions). Where  $C$  (Concept),  $R$  (Role),  $a$  and  $b$  (Individuals), these operations indicate that the Individuals apply the Concept or Role to which they refer.

As an example, let us suppose that `Employee`, `Public`, and `Private` are atomic concepts. Using the operators  $\sqcap$  and  $\sqcup$  and  $\neg$  of concepts, we can describe the concept of “Employees that are not public employees” in a Public Governance domain by the expression:

$$\text{Employee} \sqcap \neg \text{Public}$$

Similarly, we can describe “Public and Private” by the expression:

$$\text{Public} \sqcup \neg \text{Private}$$

Lastly, we can describe that “All employees have an ID” by the expression:

<sup>1</sup>In ontologies, concepts are classes that represent sets of individuals [45].

$\forall hasID.Employee$

A Knowledge Base in a DL contains two parts: *TBox* (terminology part) and *ABox* (assertional part). The *TBox* contains sentences that describe concepts and hierarchies (i.e. relationships between concepts). The *ABox* are assertions on individual objects (instance assertions) [46]. A typical assertion in the *ABox* is the one stating that an individual is an instance of a certain concept (`fernando:Man`) or assertions between individuals (`hasSon(alderic, mary)`).

A common reasoning task in DL ontologies is consistency detection [47]. For example, a standard approach to testing whether `fernando` is a member of the concept `Man` requires testing whether adding the statement (`fernando:  $\neg$ Man`) makes the *ABox* inconsistent.

**iALC** is a language based on descriptive logic that uses Atomic concepts, Logical connectives, and Complement operators to represent and reasoning about ontologies in *norm systems*. Basically, *iALC* is the intuitionistic variation of ALC<sup>2</sup>.

Until the development of this thesis, some work involving *iALC* was developed. In the paper [48] the authors present essays on the use of logical deductions as a kind of intermediate structure to assist in the task of explaining legal sentences based on legal systems in Civil Law states. In this article, the objective was to present the applicability of *iALC* in legal reasoning processes. The authors present two practical applications: first, they use a natural deduction system for *iALC* [7], based on the sequent calculus introduced in [6]. In this case, they show a way of formalizing and reasoning in order to answer the question of the Brazilian OAB Exam. The second shows another way of formalizing the usage of *iALC*, making more explicit the legal individualization of the question itself via Kripke semantics. The *iALC* semantics does not just use function over truth values, it makes use of model theory that can be given through Kripke semantics. The application of this semantics is similar to the application of world semantics in modal logic<sup>3</sup>.

Another important paper for this thesis is [50], where the authors discuss Jurisprudence and Intuitionism for investigating a logical basis for representing *Legal Reasoning* (LR) in the context of AI. They suggest the way in which Legal Reasoning can be conveyed is strongly related to how laws (or “the law”) are represented in terms of their Representation of

<sup>2</sup>*ALC* is an extension of *AL* language[11].

<sup>3</sup>Modal Logics are classical propositional logics augmented with modalities for necessity, possibility, obligations, provability, belief [49]

Legal Knowledge. Thus, *Legal Reasoning* is strongly interconnected with the *Representation of Legal Knowledge*. That is, with the chosen Legal Ontology, in the broadest sense of the term Ontology. In this case, the authors realized that Legal Reasoning cannot be based only on logic but can also be based on ontological commitments. This ontological commitment of Legal Reasoning should be guided by the underlying *jurisprudence theory* (e.g., the *Kelsenian jurisprudence*) or by judicial practice, but not by both. Therefore, *iALC* adopts legal philosophy and jurisprudence, in the tradition of Hans Kelsen's Legal Positivism where the legal system must be analyzed as a unitary and systematic entirety, constituted in a staggered manner, in which the lower norm will remove the basis of validity of an immediately higher norm. For more details on Hans Kelsen's positivism see this source [51].

For the language of *iALC*, let  $\alpha$  and  $\beta$  be concepts,  $A$  be an atomic concept,  $R$  be an atomic role,  $\delta$  be any formula, and  $x$  be a nominal. We describe *iALC* formulas by the following grammar:

$$\delta ::= \alpha \mid x : \alpha$$

where the concepts  $\alpha, \beta$  are given by the following grammar:

$$\alpha, \beta ::= A \mid \perp \mid \top \mid \neg \alpha \mid \alpha \sqcap \beta \mid \alpha \sqcup \beta \mid \alpha \multimap \beta \mid \exists R. \alpha \mid \forall R. \alpha$$

A constructive interpretation of *iALC* is a structure  $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  consisting of a non-empty set  $\Delta^{\mathcal{I}}$  of entities in which each entity represents a legal individual (a valid legal statement); a refinement pre-ordering  $\preceq$  on  $\Delta^{\mathcal{I}}$ , i.e. a reflexive and transitive relation (a pre-order), and an interpretation function  $\cdot^{\mathcal{I}}$  mapping each role name  $R \in N_R$  to a binary relation  $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$  and atomic concept  $A \in N_C$  to a set  $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$  which is closed under refinement, i.e.  $w \in A^{\mathcal{I}}$  and  $w \preceq w'$  implies  $w' \in A^{\mathcal{I}}$ . We will also refer to this last property as the *heredity rule*, as it applies to any concept, not only the atomic ones.

The interpretation  $\mathcal{I}$  is lifted from atomic concepts to arbitrary concepts as follows:

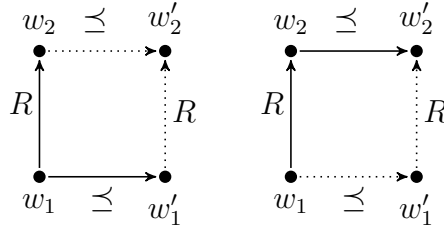
$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
\perp^{\mathcal{I}} &= \emptyset \\
(\alpha \sqcap \beta)^{\mathcal{I}} &= \alpha^{\mathcal{I}} \cap \beta^{\mathcal{I}} \\
(\alpha \sqcup \beta)^{\mathcal{I}} &= \alpha^{\mathcal{I}} \cup \beta^{\mathcal{I}} \\
(\neg \alpha)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \forall y, x \preceq y \Rightarrow y \notin \alpha^{\mathcal{I}}\} \\
(\forall R. \alpha)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \forall y (x \preceq y \Rightarrow \forall z ((y, z) \in R^{\mathcal{I}} \Rightarrow z \in \alpha^{\mathcal{I}}))\} \\
(\exists R. \alpha)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \forall y (x \preceq y \Rightarrow \exists z ((y, z) \in R^{\mathcal{I}} \wedge z \in \alpha^{\mathcal{I}}))\} \\
(\alpha \multimap \beta)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \forall y, (x \preceq y \wedge y \in \alpha^{\mathcal{I}} \Rightarrow y \in \beta^{\mathcal{I}})\}
\end{aligned}$$

The logic follows the semantics of **IK** [52], where the structures  $\mathcal{I}$  are models for *iALC* if they satisfy two frame conditions (let  $R$  be a role, and  $w_1$  and  $w_2$ , worlds):

**F1** if  $w_1 \preceq w'_1$  and  $w_1 R w_2$  then  $\exists w'_2. w'_1 R w'_2$  and  $w_2 \preceq w'_2$ ; and

**F2** if  $w_2 \preceq w'_2$  and  $w_1 R w_2$  then  $\exists w'_1. w'_1 R w'_2$  and  $w_1 \preceq w'_1$ .

We can see them as conditions for completing the following diagrams, respectively:



### 5.3

#### The Tableau System for *iALC*

In this section, the concept of semantic tableaux will be presented, its characteristics and functioning, as well as its importance in the inference process in the area of mathematical logic. The objective is to understand how this method can help in analyzing and verifying the validity of arguments and propositions. In this work, these propositions are constructions that combine to form a Knowledge Representation Base.

The semantic tableaux, also known as the tableau method, was devised independently by Evert Willem Beth (1955), Hintikka (1955), and Schütte (1956) and later developed by Smullyan (1968) [53]. Essentially, this important method is dual to Gentzen's natural deduction (1934).

It consists of building a tree of possibilities that represents all the different possible interpretations of a set of formulas for a given logic. The process of deriving a tableau comprises a set of inference rules that guide the construction of this tree of possibilities. Each node in the tree corresponds to a formula or set of formulas, and the branches represent the different interpretations of these formulas based on these rules. The objective is to check whether it is possible to close all branches of the tree consistently, that is, without generating a contradiction. In short, the tableau structure (tree) is used to explore different possibilities and determine whether a given formula is valid, satisfiable, or unsatisfiable.

To facilitate understanding of the method and specifically our reasoning method (*iALC* tableau), we will initially present the classical propositional logic (CPL) method. This form of presentation is viable because our tableau system for *iALC* will be built from elements and rules for CPL, plus some new operator-specific rules for *iALC*.

The Tableau method for classical logic is similar to the resolution method in the sense that a Tableau proof is a proof by reduction to absurdity, determining satisfiability for finite sets of formulas. The main idea of any deduction system is its calculi. In the case of semantic tableaux, their calculus is composed of a set of rules that define the transformation steps of a formula into its respective subformulas. A classic way is to represent this process as a tree structure where each node (sub-formula) has its own parent (the formula). As said before, at the end of this process, we have the ability to determine the satisfiability of a formula.

**Definition 5.1** (*Alpha and Beta Rules*). *The semantic tableau inference rules are grouped by Alpha type rules, where there is no branch bifurcation in the derivation of the formula into subformulas, and Beta type rules, where there is branch bifurcation in the derivation of the formula into subformulas. Let  $\alpha$  and  $\beta$  be two formulas of Classic Propositional Logic, the CPL calculus is shown in rules  $R_1$  to  $R_9$ .*

To clarify the application of these rules, we will prove the formula  $(P \rightarrow Q) \rightarrow (\neg Q \rightarrow \neg P)$  (an instance of the contraposition law). The first step is to *negation* of the formula:  $\neg((P \rightarrow Q) \rightarrow (\neg Q \rightarrow \neg P))$ , Fig. 5.2.

As we can see, each node in a tree represents a subformula of the formula at the origin, thus respecting the subformula property. Furthermore, for an easy treatment, we added some meta-information in each subformula. On the left side, there is a formula index between parenthesis, on the right side there are two extra-information labeled  $i : R_n$ , where  $i$  connects this sub-formula with the ancestral formula, and the  $R_n$  which represents the index of rule

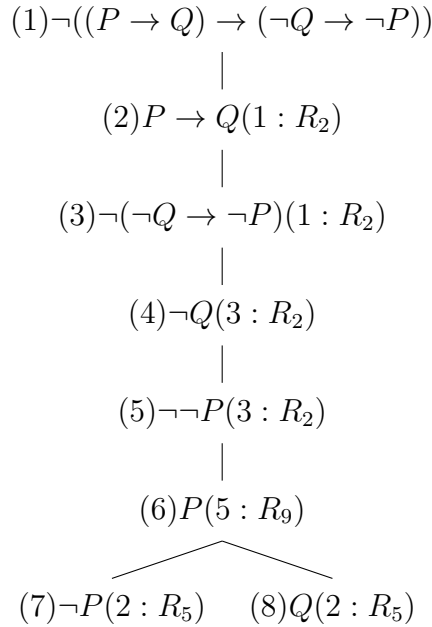
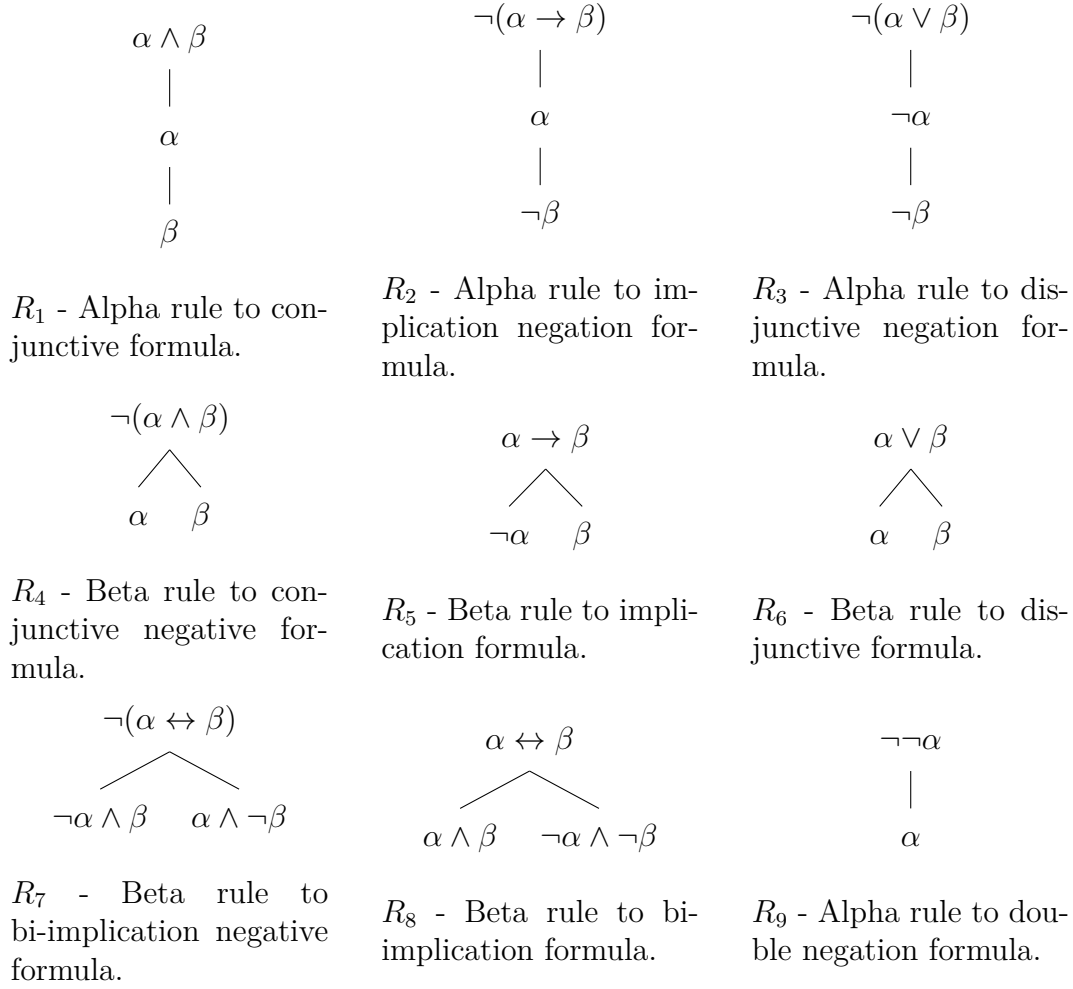


Figure 5.2: The proof tree example.

applied to the formula. This tableau is an example of the satisfiability of a formula, as the branch on the left side presents a direct contradiction  $(7)\neg P$  and  $(6)P$ , and similarly, there is a contradiction between  $(8)Q$  and  $(4)\neg Q$ .

Nowadays, it is common to represent tableaux in the Smullyan style, shown in 1968 [54]. In addition to the traditional tree format<sup>45</sup>, Smullyan represented nodes with signs (*True* or *False*). Conceptually, inference rules are very similar to general rules.

For this work, our method was inspired by the intuitionist tableau proposed by Fitting [43]. But, before presenting our method, we define some basic concepts for the tableau calculation procedure for *iALC*.

**Definition 5.2** (*Annotated Formula*). An annotated formula  $\delta$  is a pair  $\mathcal{T}(w : \alpha)$  or  $\mathcal{F}(w : \alpha)$ , such as  $\{\lambda_1(w_1 : \alpha_1), \lambda_2(w_2 : \alpha_2), \dots, \lambda_{n-1}(w_{n-1} : \alpha_{n-1}), \lambda_n(w_n : \alpha_n)\}$ , where  $w_i \in W$ .  $W$  is a nonempty set (worlds in Kripke semantics)<sup>6</sup> partially ordered by  $\preceq$ .  $\alpha_i$  is an *iALC* concept and  $\lambda_i$  is a signal of formula,  $\lambda_i \in \{\mathcal{T}, \mathcal{F}\}$ . An interpretation function  $\mathcal{I}$  over the signal of formula  $\delta$ , such that  $\mathcal{I}(\mathcal{T}) = \text{true}$  and  $\mathcal{I}(\mathcal{F}) = \text{false}$ .

**Definition 5.3** (*Relation ( $\mathcal{R}$ ) between worlds*). In the tableau, this is a binary relationship between possible worlds that captures the notion of accessibility or connection between them. The pre-order is a reflexive and transitive relation in a set of possible worlds.

**Definition 5.4** (*Pre-order Relation ( $\preceq$ )*). In the tableau, this is a binary relationship between possible worlds that captures the notion of order between them.

**Definition 5.5** (*A Tableau*). A tableau is a finite sequence of annotated formulas  $(\delta_1, \delta_2, \dots, \delta_n)$ . If  $\Upsilon$  is a tableau,  $\delta \in \Upsilon$ , and  $\delta$  derivate  $\delta'$  when applying one of the tableau expansion rules (5.3.2),  $\delta' \in \Upsilon'$ , then  $\Upsilon'$  is also a tableau.

**Definition 5.6** (*A Branch*). A branch is said as closed when, given a formula  $\varphi_i$ , both  $\varphi_i$  and its negation  $\neg\varphi_i$  are contained in the same branch. The closing of the branch will be signaled with  $(\times)$  close. Otherwise, the branch is said to be open and signaled with  $(\checkmark)$ .

**Definition 5.7** (*Closed Tableau*). A tableau is said closed if all of its branches are closed.

<sup>4</sup>A general approach to proof based on a graphical representation of branches to explore different interpretations

<sup>5</sup>There is disagreement as to who first created this graphic method, Hintikka or Smullyan [55].

<sup>6</sup>In our domain of legal norms they are VLSs.

**Definition 5.8.** Let an annotated formula  $\mathcal{T}(w : \varphi)$  or  $\mathcal{F}(w : \varphi)$ , If  $S$  is a set of annotated formulas and  $H$  is a single annotated formula, we will write  $S \cup \{H\}$  simply as  $\{S, H\}$ .

**Definition 5.9** (*Realizable Formula*). We call a set of annotated formulas realizable if there is some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$  such that  $\Gamma \models \delta_1, \dots, \Gamma \models \delta_n$ , and  $\Gamma \not\models \beta_1, \dots, \Gamma \not\models \beta_m$ , where  $\delta_{1..n}$  are annotated formulas with  $\mathcal{T}$  and, so either,  $\beta_{1..m}$  are annotated formulas with  $\mathcal{F}$

### 5.3.1

#### Related works

In the next section, we will introduce the original element of this work, a new tableau system for the logic *iALC*. This tableau system is a modified version of a proof system built by Fitting [43], and this one was originally due to Beth.

Natural Deduction is a type of deduction system that aims to represent the structure of the operators of the language with its rules. Jaśkowski [56] and Gentzen [57] developed two systems for natural deduction, which have differences in the presentation of rules but are equivalent in expressiveness: Gentzen's has tree-shaped derivations whereas Jaśkowski's produces linear derivations. In this work, we follow Gentzen's approach due to the history of *iALC* with tree-like derivations produced with Sequent Calculus.

From the initial works in tableaux, systems for different kinds of logic were created [58, 59, 60]. This section presents the related works to the system described in this chapter.

The *Pellet*, (written in Java and open source), is the first sound and complete reasoner for OWL-DL [61]. The authors mention that Pellet implements several state-of-the-art optimization techniques found in the Descriptive Logic literature. These include normalization, simplification, absorption, semantic branching, Backjumping, Caching Satisfiability Status, Top-Bottom Search for Classification, and Model Merging.

The *FaCT++* [62], is a reasoner for SHOIQ (and so OWL DL). FaCT++ uses a tableau-based calculi system too. The authors show that its architecture implements some optimizations that will be discussed in Section (5.6.1).

### 5.3.2

#### iALC Tableau Rules

Each of the rules below operates on two parts of the tableau. The top part represents the formulas that are being analyzed in the tableau construction process. Initially, the top part consists of the original formula that is being



tested for validity. During the process of applying the transition rules, the top part is modified as the formulas are decomposed, expanded, or solved. The bottom part of the tableau is where information about the analysis of the formulas is recorded. If a branch is closed due to a contradiction, this is indicated at the bottom.

$$\frac{S, \mathcal{T}(w : (A \multimap B))}{S, (w \prec w), \mathcal{F}(w : A) \mid S, (w \prec w), \mathcal{T}(w : B)} [\mathcal{T} \multimap]$$

Figure 5.3: ( $\mathcal{T}$ ) Implication (intuitionistic) rule.

$$\frac{S, \mathcal{F}(w : (A \multimap B))}{S, (w \prec w'), \mathcal{T}(w' : A), \mathcal{F}(w' : B)} [\mathcal{F} \multimap]$$

Figure 5.4: ( $\mathcal{F}$ ) Implication (intuitionistic) rule, where  $w'$  is new.

$$\frac{S, \mathcal{T}(w : (A \sqcup B))}{S, \mathcal{T}(w : A) \mid S, \mathcal{T}(w : B)} [\mathcal{T} \sqcup]$$

Figure 5.5: [ $\mathcal{T} \sqcup$ ] rule.

$$\frac{S, \mathcal{F}(w : (A \sqcup B))}{S, \mathcal{F}(w : A), \mathcal{F}(w : B)} [\mathcal{F} \sqcup]$$

Figure 5.6: [ $\mathcal{F} \sqcup$ ] rule.

$$\frac{S, \mathcal{T}(w : (A \sqcap B))}{S, \mathcal{T}(w : A), \mathcal{T}(w : B)} [\mathcal{T} \sqcap]$$

Figure 5.7: [ $\mathcal{T} \sqcap$ ] rule.

$$\frac{S, \mathcal{F}(w : (A \sqcap B))}{S, \mathcal{F}(w : A) \mid S, \mathcal{F}(w : B)} [\mathcal{F} \sqcap]$$

Figure 5.8: [ $\mathcal{F} \sqcap$ ] rule.

$$\frac{S, \mathcal{T}(w : (\exists R.C))}{S, \mathcal{T}(wRw'), \mathcal{T}(w' : C)} [\mathcal{T} \exists R]$$

Figure 5.9: [ $\mathcal{T} \exists R$ ] rule, where  $w'$  is new.

$$\frac{S, \mathcal{F}(w : (\exists R.C))}{S, \mathcal{F}(wRw), \mathcal{F}(w' : C)} [\mathcal{F} \exists R]$$

Figure 5.10: [ $\mathcal{F} \exists R$ ] rule.

$$\frac{S, \mathcal{T}(w : (\forall R.C))}{S, \mathcal{T}(wRw), \mathcal{T}(w : C)} [\mathcal{T} \forall R]$$

Figure 5.11: [ $\mathcal{T} \forall R$ ] rule.

$$\frac{S, \mathcal{F}(w : (\forall R.C))}{S, \mathcal{F}(wRw'), \mathcal{F}(w' : C)} [\mathcal{F} \forall R]$$

Figure 5.12: [ $\mathcal{F} \forall R$ ] rule, where  $w'$  is new.

$$\frac{S, \mathcal{T}((w : \neg X))}{S, \mathcal{F}(w \prec w), \mathcal{F}(w : X)} [\mathcal{T} \neg] \qquad \frac{S, \mathcal{F}((w : \neg X))}{S, \mathcal{T}(w \prec w'), \mathcal{T}(w' : X)} [\mathcal{F} \neg]$$

Figure 5.13:  $[\mathcal{T} \neg]$  rule.Figure 5.14:  $[\mathcal{F} \neg]$  rule, where  $w'$  is new.

$$\frac{S, \mathcal{T}(w : \alpha)}{S, w : \mathcal{T}(\alpha)} [\mathcal{T} \alpha] \qquad \frac{S, \mathcal{F}(w : \alpha)}{S, w : \mathcal{F}(\alpha)} [\mathcal{F} \alpha]$$

Figure 5.15:  $[\mathcal{T} \alpha]$  rule.Figure 5.16:  $[\mathcal{F} \alpha]$  rule.

$$\frac{(i, j)}{j} [Heredity] \qquad \frac{}{(i, i)} [Reflexivity]$$

Figure 5.17: Heredity rule.

Figure 5.18: Reflexivity rule.

$$\frac{(i, j)(j, k)}{i, k} [Transitivity]$$

Figure 5.19: Transitivity rule.

The rules presented are made up of basic rules about how the formulas at the top of the board are derived to the bottom. Basically, there are two main rules: *Formulas of type  $\alpha$* : The consequences of formulas of type  $\alpha$  are direct consequences, that is, they remain in the same branch and do not generate forks. *Formulas of type  $\beta$* : in this case, formulas of type  $\beta$  are not direct and, therefore, bifurcate into two distinct branches, each of which is a possibility of analyzing the given formula. As we can see (Figures: 5.3, 5.5 and 5.8), the rules for  $\beta$  (fork) formulas show their left and derived from the right-hand side and separated by a "|". This  $S$ , present in the rules, corresponds to the set of formulas that have yet to be derived (top part) and the set of formulas already derived (bottom part).

## 5.4

### Main Properties

In this Section, we show that essential properties expected of tableaux systems hold in the system proposed for *iALC*.

#### 5.4.1

##### Soundness

To reason about the various legal acts and legal rules, we need our fundamental logical inductive system to be correct, which provides us with

a criterion of logical validity of legal reasoning. Therefore, this section is responsible for presenting the soundness of our SAT solver.

Before proving soundness itself, we first provide some useful lemmas.

**Theorem 5.1.** *Let  $\delta_1, \delta_2, \dots, \delta_n$ , is an  $iALC$  tableau. If  $\delta_1$  is realizable, so is  $\delta_{1+1}$ .*

**Lemma 5.1**  $(S, \mathcal{T}(w : (\alpha \sqcup \beta)))$ . *Let  $\alpha$  and  $\beta$  be  $iALC$  formulas,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w : \alpha)\}, \{S, \mathcal{T}(w : \beta)\}, \dots\}$ .*

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $\{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \models w : (\alpha \sqcup \beta)$ . Then  $\Gamma \models (w : \alpha)$  or  $\Gamma \models (w : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{T}(w : \alpha)\}$  or  $\{S, \mathcal{T}(w : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable.

**Lemma 5.2**  $(S, \mathcal{F}(w : (\alpha \sqcup \beta)))$ . *Let  $\alpha$  and  $\beta$  be  $iALC$  formulas,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w : \alpha)\}, \{S, \mathcal{F}(w : \beta)\}, \dots\}$ .*

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $\{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \not\models w : (\alpha \sqcup \beta)$ . Then  $\Gamma \not\models (w : \alpha)$  and  $\Gamma \not\models (w : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{F}(w : \alpha)\}$  or  $\{S, \mathcal{F}(w : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable.

**Lemma 5.3**  $(S, \mathcal{T}(w : (\alpha \sqcap \beta)))$ . *Let  $\alpha$  and  $\beta$  be  $iALC$  formulas,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w : \alpha)\}, \{S, \mathcal{T}(w : \beta)\}, \dots\}$ .*

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $\{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \models w : (\alpha \sqcap \beta)$ . Then  $\Gamma \models (w : \alpha)$  and  $\Gamma \models (w : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{T}(w : \alpha)\}$  and  $\{S, \mathcal{T}(w : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable.

**Lemma 5.4**  $(S, \mathcal{F}(w : (\alpha \sqcap \beta)))$ . *Let  $\alpha$  and  $\beta$  be  $iALC$  formulas,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w : \alpha)\}, \{S, \mathcal{F}(w : \beta)\}, \dots\}$ .*

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element

is  $\{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \not\models w : (\alpha \sqcap \beta)$ . Then  $\Gamma \not\models (w : \alpha)$  and  $\Gamma \not\models (w : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{F}(w : \alpha)\}$  **and**  $\{S, \mathcal{F}(w : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable.

**Lemma 5.5**  $(S, \mathcal{F}(w : \neg\alpha))$ . Let  $\alpha$  an iALC formula,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : \neg\alpha)\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w' : \alpha)\}, \dots\}$ .

**Proof** Since  $\delta_i$  is realizable, and it suffices to consider the case that  $\{S, \mathcal{F}(w : \neg\alpha)\}$  is the realizable element. Then there is a model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and a  $\Gamma \in \Delta^{\mathcal{I}}$  such that  $\Gamma$  realizes  $S$  and  $\Gamma \not\models (w : \neg\alpha)$ . Since  $\Gamma \not\models (w : \neg\alpha)$ , for some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma \models \delta$ . But clearly, if  $\Gamma$  realizes  $S$ ,  $\Gamma^*$  (all  $\Gamma$ ) realizes  $w'$  (for a new  $w$ ) and  $[w \prec w']$  by definition (5.4). Hence  $\Gamma^*$  realizes  $\{S, [\mathcal{T}(w \prec w'), \mathcal{T}(w' : \alpha)]\}$  and  $\delta_{i+1}$  is realizable.

**Lemma 5.6**  $(S, \mathcal{T}(w : \neg\alpha))$ . Let  $\alpha$  an iALC formula,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : \neg\alpha)\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w' : \alpha)\}, \dots\}$ .

**Proof** Since  $\delta_i$  is realizable, and it suffices to consider the case that  $\{S, \mathcal{T}(w : \neg\alpha)\}$  is the realizable element. Then there is a model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and a  $\Gamma \in \Delta^{\mathcal{I}}$  such that  $\Gamma$  realizes  $S$  and  $\Gamma \models (w : \neg\alpha)$ . Since  $\Gamma \models (w : \neg\alpha)$ , for some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma \models \delta$ . But clearly, if  $\Gamma$  realizes  $S$ ,  $\Gamma^*$  (all  $\Gamma$ ) realizes  $w'$  and  $[w \prec w']$ , by definition (5.4), then  $\Gamma^*$  realizes  $\{S, [\mathcal{F}(w \prec w'), \mathcal{F}(w' : \alpha)]\}$ , that is,  $\delta_{i+1}$  is realizable.

**Lemma 5.7**  $(S, \mathcal{T}(w : \alpha))$ . Let  $\alpha$  an iALC formula,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : \alpha)\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(\alpha)\}, \dots\}$ .

**Proof** Since  $\delta_i$  is realizable, and it suffices to consider the case that  $\{S, \mathcal{T}(w : \alpha)\}$  is the realizable element. Then there is a model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and a  $\Gamma \in \Delta^{\mathcal{I}}$  such that  $\Gamma$  realizes  $S$  and  $\Gamma \models \alpha$ . In either case,  $\delta_{i+1}$  is realizable.

**Lemma 5.8**  $(S, \mathcal{F}(w : \alpha))$ . Let  $\alpha$  an iALC formula,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : \alpha)\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(\alpha)\}, \dots\}$ .

**Proof** Since  $\delta_i$  is realizable, and it suffices to consider the case that  $\{S, \mathcal{F}(w : \alpha)\}$  is the realizable element. Then there is a model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and a  $\Gamma \in \Delta^{\mathcal{I}}$  such that  $\Gamma$  realizes  $S$  and  $\Gamma \not\models \alpha$ . In either case,  $\delta_{i+1}$  is realizable.

**Lemma 5.9**  $(S, \mathcal{T}(w : (\alpha \multimap \beta)))$ . Let  $\alpha$  and  $\beta$  be iALC formulas,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : (\alpha \multimap \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w : \alpha)\}, \{S, \mathcal{T}(w : \beta)\}, \dots\}$ .

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $S, \mathcal{T}(w : (\alpha \multimap \beta))$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $S, \mathcal{T}(w : (\alpha \multimap \beta))$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $S, \mathcal{T}(w : (\alpha \multimap \beta))$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \models w : (\alpha \multimap \beta)$ .

Then  $\Gamma \not\models (w : \alpha)$  and  $\Gamma \models (w : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{F}(w : \alpha)\}$  and  $\{S, \mathcal{T}(w : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w \preceq w$ , for some  $w$  in the branch ( $w \in W$  occur in  $S$ ).

**Lemma 5.10**  $(S, \mathcal{F}(w : (\alpha \rightarrow \beta)))$ . Let  $\alpha$  and  $\beta$  be *iALC* formulas,  $w \in W$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : (\alpha \rightarrow \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w' : \alpha), \mathcal{F}(w' : \beta)\}, \dots\}$ .

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $S, \mathcal{F}(w : (\alpha \rightarrow \beta))$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $S, \mathcal{F}(w : (\alpha \rightarrow \beta))$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $S, \mathcal{F}(w : (\alpha \rightarrow \beta))$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \not\models w : (\alpha \rightarrow \beta)$ . Then  $\Gamma \models (w' : \alpha)$  and  $\Gamma \not\models (w' : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{T}(w' : \alpha)\}$  and  $\{S, \mathcal{F}(w' : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w \preceq w'$ , for new  $w$  in the branch ( $w' \in W$  does not occur in  $S$ ).

**Lemma 5.11**  $(S, \mathcal{T}(w : (\exists R.C)))$ . Let  $w \in W$ , how defined in (5.2).

**Proof** if  $S, \mathcal{T}(w : (\exists R.C))$  is realizable, and if  $w', w' \in W$  does not occur in  $S$ , then  $S, \mathcal{T}(w' : C)$  is realizable.

**Lemma 5.12**  $(S, \mathcal{F}(w : (\exists R.C)))$ . Let  $w \in W$ , how defined in (5.2).

**Proof** if  $S, \mathcal{F}(w : (\exists R.C))$  is realizable, and if  $w', w' \in W$  occur in  $S$ , then  $S, \mathcal{F}(w' : C)$  is realizable.

**Lemma 5.13**  $(S, \mathcal{T}(w : (\forall R.C)))$ . Let  $w \in W$ , how defined in (5.2).

**Proof** if  $S, \mathcal{T}(w : (\forall R.C))$  is realizable, and if  $w', w' \in W$  occur in  $S$ , then  $S, \mathcal{T}(w' : C)$  is realizable. Similar to lemma 5.12.

**Lemma 5.14**  $(S, \mathcal{F}(w : (\forall R.C)))$ . Let  $w \in W$ , how defined in (5.2).

**Proof** if  $S, \mathcal{F}(w : (\forall R.C))$  is realizable, and if  $w', w' \in W$  does not occur in  $S$ , then  $S, \mathcal{F}(w' : C)$  is realizable. Similar to lemma 5.11.

**Theorem 5.2** (Soundness). Let  $\delta$  be a valid formula of *iALC*,  $\Gamma$  a set of *iALC* formulas, and all branches of the tableaux are closed. Then,  $\Gamma \vdash_{tab} \delta$  implies  $\Gamma \models_{tab} \delta$ .

**Proof** If  $\delta$  is derived from some rule in the *iALC* tableau, then  $\delta$  is true in all possible models.

## Reasoning in *iALC* tableau calculus

## 5.6

### The KRM Machine

The kernel of this unit was developed in Java language. This choice is the ease of implementing computational algorithms capable of representing graph [63] structures, as well as the main graph search engines. `LinkedList` structures available in the environment facilitated the development of structural representations of our tableau [64]. We represent the tableau formulas as nodes in a graph and their adjacency was made easier with the representation of this linked list. Building Java objects with the representations present in *iALC* formulas. For example, `id_node`, formula sign, type of rule to apply (alpha or beta), the world of the formula, order relations, main connective, etc.

For the lexical and syntactic analyzer of the *iALC* logical language, a grammar (Fig. 5.21) and its parser with the ANLTR4 (ANother Tool for Language Recognition) [65] version 4 were developed.

Here are some reasons why we adopted ANTLR4 in this project:

- *Java code*: the analyzers are generated in Java language. The same language we used to build Tableau *iALC*, thus allowing full integration with the project. In practice, the parser was integrated as a package into the project.
- *Ease of use*: ANTLR provides an intuitive and easy-to-learn grammar syntax based on Extended Backus–Naur Form [66], which makes defining language grammars more accessible.
- *Active community support*: Due to its simplicity, ANTLR has an active community of developers. Providing an abundance of tutorials available to support projects.

We take advantage of this grammar and extend it to an attribute grammar, see Appendix A. The idea was to assign semantic meaning to the nodes of the Abstract Syntactic Tree (AST) generated by this parser. In this case, our interest was in identifying the main symbol of a subformula in the tableau calculi (main element of KRM). With this, given a formula, alpha or beta type rule, and a visit mechanism implemented as an interface in ANLTR, We can divide a formula into subformulas through its main symbol identified in the visit (path in this AST) of its root node, the main symbol of the formula.

$\langle S \rangle$	$\models$	$\langle \text{Formula} \rangle$
$\langle \text{Formula} \rangle$	$\models$	$\langle \text{LPAREN} \rangle \langle \text{CHARACTER} \rangle \langle \text{RPAREN} \rangle$
$\langle \text{Formula} \rangle$	$\models$	$\langle \text{Formula} \rangle \langle \text{binConnective} \rangle \langle \text{Formula} \rangle$
$\langle \text{Formula} \rangle$	$\models$	$\langle \text{LPAREN} \rangle \langle \text{Formula} \rangle \langle \text{RPAREN} \rangle$
$\langle \text{Formula} \rangle$	$\models$	$\langle \text{NOT} \rangle \langle \text{Formula} \rangle$
$\langle \text{Formula} \rangle$	$\models$	$\langle \text{FORALL} \rangle \langle \text{ROLE} \rangle " . " \langle \text{LPAREN} \rangle \langle \text{CHARACTER} \rangle \langle \text{binConnective} \rangle \langle \text{CHARACTER} \rangle$
$\langle \text{RPAREN} \rangle$		
$\langle \text{Formula} \rangle$	$\models$	$\langle \text{EXISTS} \rangle \langle \text{ROLE} \rangle " . " \langle \text{LPAREN} \rangle \langle \text{CHARACTER} \rangle \langle \text{binConnective} \rangle \langle \text{CHARACTER} \rangle$
$\langle \text{RPAREN} \rangle$		
$\langle \text{Formula} \rangle$	$\models$	$\langle \text{MODAL} \rangle$
$\langle \text{ROLE} \rangle$	$\models$	$\langle \text{CHARACTER} \rangle$
$\langle \text{LPARAN} \rangle$	$\models$	" ( "
$\langle \text{RPARAN} \rangle$	$\models$	" ) "
$\langle \text{MODAL} \rangle$	$\models$	$\langle \text{FORALL} \rangle \mid \langle \text{EXISTS} \rangle$
$\langle \text{FORALL} \rangle$	$\models$	" $\forall$ "
$\langle \text{EXISTS} \rangle$	$\models$	" $\exists$ "
$\langle \text{binConnective} \rangle$	$\models$	$\langle \text{CONJ} \rangle \mid \langle \text{DISJ} \rangle \mid \langle \text{iIMPL} \rangle$
$\langle \text{CONJ} \rangle$	$\models$	" $\sqcap$ "
$\langle \text{DISJ} \rangle$	$\models$	" $\sqcup$ "
$\langle \text{iIMP} \rangle$	$\models$	" $\longrightarrow$ "
$\langle \text{NOT} \rangle$	$\models$	" $\neg$ "
$\langle \text{CHARACTER} \rangle$	$\models$	$[0-9a-zA-Z]$
$\langle \text{ENDLINE} \rangle$	$\models$	$(\backslash r \mid \backslash n)^+$
$\langle \text{WHITESPACE} \rangle$	$\models$	$(" " \mid \backslash t)^+$

Figure 5.21: eBNF for *iALC*

### 5.6.1

#### The KRM Kernel

In this section, we will present the main algorithms of our solver. As seen in the previous section, the KRM kernel was developed in Java language and some abstractions that will be shown here (Algorithm 1) have their respective implementation in structures already discussed in Section (5.6).

The precondition to start the process is a formula *iALC* that represents the auditing criteria. In the (Algorithm 1), line 1 is a constructor for main structures. Remember that we represent formulas by node objects. Thus, a `LinkedList<Node>` was fundamental for this implementation, this structure allowed us to create connections between nodes automatically. In line 2, there is an assignment to `mainLinkedList` with the new formula derivate of the original formula. In practice, this new formula is the original formula with world, signal, and check flag<sup>7</sup> increments. Initially, this formula starts with a false check flag value; The algorithm's stopping criterion is a full set of checked

<sup>7</sup>The check flag is a boolean Node type attribute. The formulas that had to be visited are checked with true values. Conversely, new and unvisited formulas are marked as false;



derived formulas in the main structure, lines 3-4. Line 5, shows the parser tree visitor over the `LocalFormula`, a formula that was chosen to analyze. In this step, an *abstract syntax tree* (AST) [65] was used to highlight the main symbol of the formula. Line 6 adds this formula to the sheet structure list. This list will help in the process of inserting derived formulas as children of the original formula into a specific branch. A *depth-first search* (DFS) [67] is used to search for these leaves of the node being analyzed. Line 7, recover if the formula is  $\alpha$  or  $\beta$ . If the formula is  $\alpha$ -type, it starts a loop (size of `leavesList`) that creates new formulas by applying rules. In line 10, the original formula is split into two new formulas. For these, all Node type attributes were applied (lines 11-15). An interesting point is the definition of the *world* of this new formula (line 11). The process here involves choosing a world (current or new) according to the criteria presented in the formulas shown in Section (5.3.2). If the reader wants to understand this process, the Algorithm 2 shows this important step.

Lines 17-24 are similar to lines (9-15) already presented.

Finally, 26-30 are decidability instructions.

This algorithm is just a high-level view to demonstrate the processes involved in implementing our SAT solver. If the reader wishes to see the implementation of this algorithm, it is available in the [GitHub Link](#).

The reader can see the soundness proof of this algorithm in Appendix D.

*The algorithm that decides what world is applied:*

Change of world rules is used in modal tableaux to explore different scenarios (possible worlds) in relation to the modal relationships specified in the problem. In an intuitionist tableau shown by Fitting [43], the rules of  $\mathcal{F} \rightarrow$ ,  $\mathcal{F} \neg$ ,  $\mathcal{F} \forall$ , and  $\mathcal{T} \exists$ , to exclude all formulas (signed  $\mathcal{F}$ ) in that branch<sup>8</sup>. In our *iALC* rules, we create new worlds (not yet present in the tableau). Thus, for our tableau, the world change rules allow the construction of possible states of the world, checking and validating the formula in different scenarios. This is important in relation to the hierarchical system of legal norms, as it allows us to examine how relationships between worlds (laws) behave in different contexts.

Before explaining how the algorithm decides which world will be applied, it is necessary to define the concept of the complement of the formula.

**Definition 5.10** (*Complement of formula*). *The complement of a formula is any formula that is syntactically the same but with a different annotation*

<sup>8</sup>To maintain coherence with the more restricted view of negation in intuitionism

---

**Algorithm 1** The tableau-based satisfiability algorithm for iALC

---

**Require:** *formula*

```

1: mainLinkedList :: LinkedList<Node>
2: mainLinkedList<node>[Formula, AssignFormula(Formula)]
3: while (unchecked) do
4:   LocalFormula  $\leftarrow$  searchUnchecked(mainLinkedList)
5:   mainBinToken  $\leftarrow$  (parseTreeVisitor(LocalFormula))
6:   leavesList  $\leftarrow$  DFSLeafList(LocalFormula)
7:   AlphaBeta  $\leftarrow$  SAlphaBeta(LocalFormula)
8:   if AlphaBeta == "Alpha" then
9:     for l = 0 to leavesList.size() - 1 do
10:      NewFormula', NewFormula''  $\leftarrow$  SplitAlpha(LocalFormula)
11:      objN.world(NewFormula', NewFormula'')  $\leftarrow$ 
        rules(LocalFormula)
12:      objN.sign(NewFormula', NewFormula'')  $\leftarrow$ 
        rules(LocalFormula)
13:      mainLinkedList.update  $\leftarrow$  check(LocalFormula)
14:
15:      ...  $\triangleright$  Other definitions for the Node object (objN).
16:     end for
17:   else if AlphaBeta == "Beta" then
18:     for l = 0 to leavesList.size() - 1 do
19:      NewFormula', NewFormula''  $\leftarrow$  SplitBeta(LocalFormula)
20:      objN.world(NewFormula', NewFormula'')  $\leftarrow$ 
        rules(LocalFormula)
21:      objN.sign(NewFormula', NewFormula'')  $\leftarrow$ 
        rules(LocalFormula)
22:      mainLinkedList.update  $\leftarrow$  check(LocalFormula)
23:      ...  $\triangleright$  Other definitions for the Node object (objN).
24:     end for
25:   end if
26:   if findUnchecked() then
27:     unchecked = true
28:   else
29:     unchecked = false
30:   end if
31: end while

```

---

(signal), as shown in Definition (5.2).

---

**Algorithm 2** World definition algorithm
 

---

```

1: procedure WRLDEFINITION(node)      ▷ The node visited in the graph.
2:   branch  $\leftarrow$  (SBranch(node))
3:   cNode  $\leftarrow$  (SComp(node, branch))
4:   if (cNode.world == node.world) then
5:     CloseBranch(node);
6:   else if (cNode.world != Node.world) then
7:     w  $\leftarrow$  (SRelation(node.world, cNode.world, branch))
8:     if (w) then
9:       applyWorld(node);
10:      CloseBranch(node);
11:    end if
12:  end if
13: end procedure

```

---

The Algorithm (2) is a procedure with a *node* parameter (line 1). The *node* is a formula that is being visited in the tableau graph. In line 3, the complement of *node* is found in the same branch (*cNode*).

If the world of the complement node and the world of the current node are different (line 6), then the next line is available. Thus, line 7 searches for a relation between the current *world* of *node*, the *world* of its complement, and the branch. If the return is *true*, then the world will be applied definitively to the *node*, and the branch will be closed.

We emphasize that for this thesis, our algorithm does not address issues related to complexity in tableaux. In the literature [11], we find two main sources of complexity in tableaux calculi: *or-branching* and *and-branching*. We suggest reading this paper [68] where the authors present a hybrid system of hypertableau and resolution, for description logic, which deals with these complexities.

### 5.6.2

#### The Model and Counter-model Logs File

Remember, an *iALC model* is an interpretation that satisfies all formulas in the tableau branch, taking into account the rules of *iALC* tableau. In other words, a model is a structure that assigns true or false values to the formula, according to the restrictions imposed by *iALC* tableau calculi, and makes all formulas in the branch true. In a complementary way, an *iALC counter-model* is an interpretation that satisfies all of the formulas present in the branch, except one. This means that there is a formula in the branch that is not true under the interpretation given by the counter-model.

$\langle S \rangle$	$\models$	$\langle \text{Log} \rangle$
$\langle \text{Log} \rangle$	$\models$	$\langle \text{LEFTC} \rangle " == > : " \langle \text{RIGHTC} \rangle$
$\langle \text{LEFTC} \rangle$	$\models$	$\langle \text{LPAREN} \rangle \langle \text{ID} \rangle \langle \text{RPAREN} \rangle \langle \text{WORLD} \rangle \langle \text{SIGN} \rangle \langle \text{CONCEPT} \rangle$
$\langle \text{RIGHTC} \rangle$	$\models$	$\langle \text{LPAREN} \rangle \langle \text{ID} \rangle \langle \text{RPAREN} \rangle \langle \text{WORLD} \rangle \langle \text{SIGN} \rangle \langle \text{CONCEPT} \rangle$
$\langle \text{LPARAN} \rangle$	$\models$	$" ( "$
$\langle \text{RPARAN} \rangle$	$\models$	$" ) "$
$\langle \text{WORLD} \rangle$	$\models$	$"_w" [ ' ] *$
$\langle \text{ID} \rangle$	$\models$	$[ 0 - 9 ] +$
$\langle \text{SIGN} \rangle$	$\models$	$"T" \mid "F"$
$\langle \text{CONCEPT} \rangle$	$\models$	$[ 0 - 9 a - z A - Z ] +$

Figure 5.22: eBNF for counter-model log file.

The log file is the result of the process of extracting models and counter-models present in the branches. As a result, this counter-model file is used as input to a transpiler<sup>9</sup> contained in the KRM Module. To understand how this log file is used for the transpiler, we show in Fig 5.22 a grammar for this. In the next section, we will detail this step of transforming the counter-model into SPARQL query.

### 5.6.3

#### From Counter-model to SPARQL

The last activity of the KRM module is to obtain the counter-models inferred from the KRM Machine and transform them into SPARQL queries. For this process, an *iALC* analyzer will perform this transformation through a process that maps the atomic concepts and roles of *iALC*, available in the counter-model, and transforms them into query structures in SPARQL language.

Our interest is in identifying possible situations of non-conformity in the knowledge base. We look at the result of the SPARQL query to see if such a property is found. In fact, as we are executing a query transformed from a counter-model, the following condition must be analyzed:

***If the query result returns some value, then we find a non-conformity situation. Otherwise, for this domain (law), the base is in compliance.***

In cases where there is compliance for all domains (laws), we can assign a quality certificate to this knowledge base.

<sup>9</sup>It is a generic term used to describe the process of converting code from one language to another [69].

For a better understanding, let's start by remembering the basic concepts of an RDF structure and SPARQL language.

The RDF (*Resource Description Framework*) is a data model (metadata) used to represent information to facilitate the search for resources on the web [70].

According to [71, 72], the RDF increment the structure of *links* of the web when using *URIs* to appoint the relationship between the web features, available in the form of a triple element  $\langle \text{subject} \rangle$ ,  $\langle \text{predicate} \rangle$  and  $\langle \text{object} \rangle$ . Consequently, this model RDF forms a graph (RDF graph) directed and labeled where the most external nodes represent resources (subject or object) that are related by a predicate (edge) [73]. Resources can be seen as any information, such as a document, a person, etc. In this case, each feature is assigned an identifier element *IRI* (*Internationalized Resource Identifier*). Figure 5.23 represents the RDF model primitives and will be used to present the necessary mapping in constructing SPARQL queries on an RDF knowledge base.

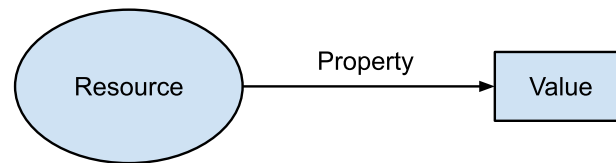


Figure 5.23: RDF Model Primitives.

To exemplify, we present two figures that represent instances of our  $KB_{RDF}$ . In these cases, both the code listings 8 and 9 serialize a single employee. Technically, in terms of XML/RDF standards<sup>10</sup>, the *Resource* (identified by the `rdf:nodeID` attribute) is the one given by the identifier (`N7654fa3513984ea89c0ade256a971939`). The *Predicate*, the properties of the RDF, are identified by the elements within the RDF node: `FOAF:cpf`, `FOAF:tipoCargo`, `FOAF:simbolo`, eg. Finally, the *Value*, which are the values associated with each property. In this case, for the property `FOAF:nome`: FERNANDO ANTONIO DANTAS GOMES PINTO.

Listing 5.1 shows how we can define a SPARQL query that extracts information about this XML/RDF file.

```

1 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2 SELECT ?nome ?matricula ?portaria
3 WHERE {

```

<sup>10</sup>For more general examples, the reader can consult the XML/RDF specification at [74].

**Listing 8** *RDF/XML* data for Commissioned Position.

---

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
>
  <foaf:Funcionario rdf:nodeID="N7654fa3513984ea89c0ade256a971939">
    <foaf:cpf>922.355.544-12</foaf:cpf>
    <foaf:tipoCargo>CC</foaf:tipoCargo>
    <foaf:simbolo>DAS-06</foaf:simbolo>
    <foaf:dataEfeito>2017-01-01</foaf:dataEfeito>
    <foaf:matricula>60/210917-1</foaf:matricula>
    <foaf:cargo>ASSISTENTE I</foaf:cargo>
    <foaf:nome>FERNANDO ANTONIO DANTAS GOMES PINTO</foaf:nome>
  </foaf:Funcionario>
```

---

**Listing 9** *RDF/XML* data for Função Gratificada.

---

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
>
  <foaf:Funcionario rdf:nodeID="N7654fa3584ea89c0ade">
    <foaf:cpf>922.355.544-12</foaf:cpf>
    <foaf:tipoCargo>FG</foaf:tipoCargo>
    <foaf:simbolo>FG-01</foaf:simbolo>
    <foaf:dataEfeito>2018-02-01</foaf:dataEfeito>
    <foaf:matricula>60/210917-1</foaf:matricula>
    <foaf:cargo>FUNCAO AUX I</foaf:cargo>
    <foaf:nome>FERNANDO ANTONIO DANTAS GOMES PINTO</foaf:nome>
  </foaf:Funcionario>
```

---

```
4 [rdf:nodeID="N7654fa3584ea89c0ade"] foaf:nome ?nome ;
5                                     foaf:matricula ?matricula ;
6                                     foaf:portaria ?portaria .
7 }
```

Code Listing 5.1: Query for the specific instance (Listing 9).

In this query, PREFIX foaf: <http://xmlns.com/foaf/0.1/> defines the namespace foaf with the aim of simplifying references in properties. SELECT ?nome ?matricula ?portaria specifies that we are interested in the values of properties foaf:nome, foaf:matricula and foaf:portaria (projection variables). The [rdf:nodeID="N7654fa3584ea89c0ade"], selects the subject for the node with the specified identifier. Finally, the foaf:nome ?nome ; foaf:matricula ?matricula ; foaf:portaria ?portaria extracts the corresponding values of properties for the specified subject. This query returns the nome, matrícula, and portaria values for the subject identi-

fied by the node with the identifier to the subject identified by node with id N7654fa3584ea89c0ade.

*The mapping:*

Finally, we present Table 5.1 that shows the rules for mapping the information extracted from the counter-model file. As seen, we are concerned with treating only the concepts and roles extracted from the formulas and directly building them as elements RDF. The transpiler uses the grammar of the counter-model file and, based on these mapping rules, transforms it into SPARQL query.

In practice, the *iALC* roles can be mapped to *RDF* properties, and *iALC* nominal “concepts” to *RDF* object “values”. Then, from RDF to SPARQL is also a direct transformation.

An observation about the Role mapping rule is that it is performed by an indirect extraction by the counter-model file. In this, from the node “ID” (see the grammar in Figure. 5.22) we can look for the branch until it reaches its ancestor that contains the information about its role (Listing 10). This task is performed as a subprocess associated with the transpiler.

---

**Listing 10** Looking for role in formula.

---

```
private static String getRole(Node token) {
    String predicate=null;
    int idNo = token.getIdIndex();
    for (Node tokensMainList : PrincipalNew.mainList) {
        if (tokensMainList.getIdIndex() == idNo){
            String idPai = tokensMainList.getIdIndexPai();
            for (Node paiMainList : PrincipalNew.mainList) {
                if (paiMainList.getIdIndex() == Integer.parseInt(idPai)){
                    String regex = "[ ]+([a-zA-Z]+)";
                    Pattern pattern = Pattern.compile(regex, Pattern.DOTALL);
                    Matcher matcher = pattern.matcher(paiMainList.getFormula());
                    while (matcher.find()) {
                        predicate = matcher.group(1);
                    }
                }
            }
        }
    }
    return predicate;
}
```

---

Table 5.1: Some general guidelines for transforming *iALC* to a SPARQL query.

Mapping Rules			Example			
Maps	iALC (from)	RDF Primitives (to)	iALC Formula	Mapping	Parser	RDF
Concepts	Atomic Concepts	RDF Object (value)	XEmployeeAt.Sempma	Sempma	$\mapsto$	Sempma
Roles	Roles	RDF Properties	XEmployeeAt.Sempma	EmployeeAt	$\mapsto$	EmployeeAT



*The SPARQL query templates:*

At this stage of the research, we used three basic templates that helped construct SPARQL queries. The first template is applied according to the number of open branches in the tableau. To do this, we start the construction process by associating each branch with a single complete SPARQL structure (SELECT clause, *projection variables* and WHERE clause). For example, if there are two counter-examples generated by Tableau, then this template will be used twice. In other words, we will have a query for each open branch.

The second template is applied where the counter-model only presents signed with  $\mathcal{T}$ , then the query is constructed by applying these variables to each WHERE clause of the subset of queries presented in the first template.

Finally, the third template is for cases where the counter-model presents variables signed with  $\mathcal{F}$ , so the query is built using the FILTER NOT EXISTS in the parameters of the WHERE clause. Then, this template is associated with the rules of the first and second templates.

Here, it is worth mentioning that, for simplicity reasons, we are defining the “name” projection variable as the default.

#### 5.6.4

##### The KRM Solver Application

In this subsection, we will present a small sample of the ability of our SAT Solver to present the formula solution tree in *iALC* as well as the LOG files containing information about models and counter-models of this process. Furthermore, we will present the Log file containing the generated SPARQL query. For this, we will use the tableau presented in subsection 5.5, for the formula:

$$\vdash_{iALC} \forall temCargo.CC \rightarrow (\exists temCargo.CC \wedge \neg \exists temCargo.FG).$$

As we can see in the figure, this table contains some meta information. On the left side of each formula there is an index that uniquely identifies each derived subformula. On the right side, there is the index that points to the parent formula of this subformula. On this same side (rightmost), there are indicators of precedence relationship derived between worlds ( $w^m \preceq w^n$ ) and relations between worlds ( $w^m R w^n$ ). The last information is about the closed branch. In the left branch, there is an X (5) representing that formula (8) has its complement with formula (5), thus closing this branch.

The Log File in Listing 11 showed that the concept CC is T (true) in node five (5) and F (false) in node eight (8). closing the leftmost branch

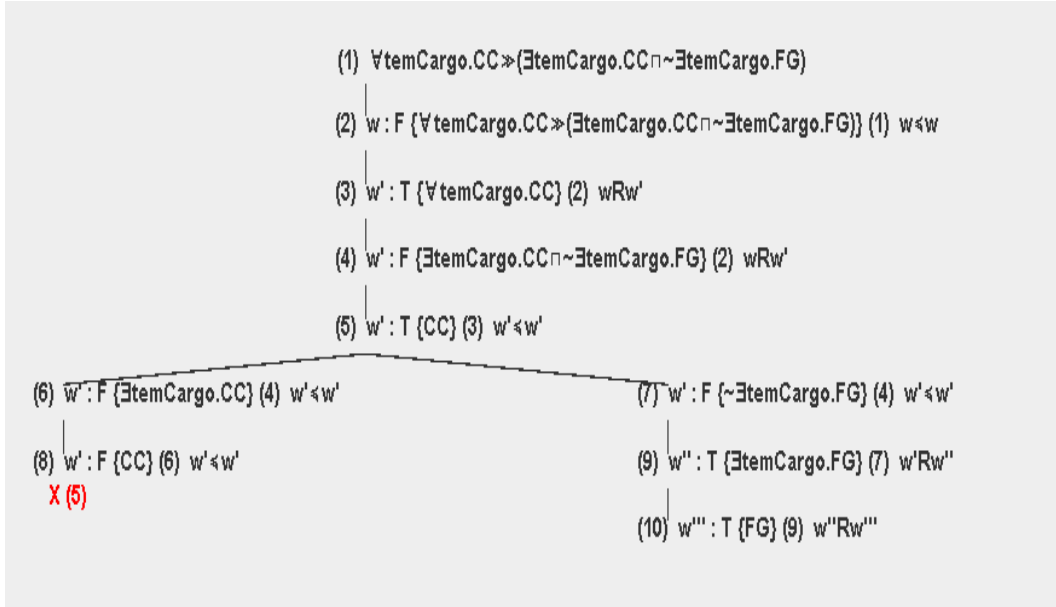


Figure 5.24: Proof tree generated by KRM SAT Solver.

of the tableau. They are both in the same world ( $w'$ ). Therefore, the [Relation==>] information, in this case, doesn't show any additional information about the relationships.

---

**Listing 11** Model Log generated by KRM SAT solver.

---

(5)  $w' \quad T \quad CC \Rightarrow :$  (8)  $w' \quad F \quad CC \quad [Relation==>]$

---

The Log File in Listing 12 showed that the rightmost branch is the counter-model. For it, the Node ten (10) contains the formula FG signed with T (true), in the world ( $w'''$ ) and the Node (5) five contains the formula CC signed with T (true), in the world ( $w'$ ).

---

**Listing 12** Counter-model Log generated by KRM SAT solver.

---

(10)  $w''' \quad T \quad FG \Rightarrow :$  (5)  $w' \quad T \quad CC$

---

Finally, we show the Listing 13, the Log file that contains the query generated by KRM SAT solver correspondent the counter-model.

---

**Listing 13** SPARQL query generated by KRM SAT Solver.

---

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
  {
    SELECT ?name
    WHERE {
      ?x foaf:temCargo 'FG' .
      ?x foaf:name ?name .
    }
  }
  {
    SELECT ?name
    WHERE {
      ?x foaf:temCargo 'CC' .
      ?x foaf:name ?name .
    }
  }
}
```

---

## 6

## Evaluation and Results

In this chapter, we evaluate and present results that demonstrate the efficiency of our approach. To do this, we carried out different *queries* (CQs<sup>1</sup> in *iALC* language) to an RDF knowledge base with the aim of verifying the ability of our architecture to support the certification process of this KB. Our evaluation criteria followed the approach used in the CQ inference and validation process for the CoreACQ<sup>2</sup> tool.

Using Competency Questions as an ontology engineering process plays a fundamental role in the ontology development life cycle. It serves as the bedrock for identifying and defining requirements. By formulating and answering CQs, ontology developers can gain a comprehensive understanding of the domain, its concepts, relationships, and constraints, which in turn guides the ontology modeling process [77].

We can define the CQ as  $\langle C, \sigma \rangle$  such that  $C$  is a question expressed in *iALC* language and  $\sigma$  is an answer to this question expressed counterexamples from the *iALC* tableau. Roughly speaking, the process of obtaining the CQs dataset was through consultations with experts and modeling based on laws published in official gazettes. We defined a set of questions focused specifically on the terminological part (TBox). This dataset can be as a gold standard for testing and benchmarking, research into logic, and tool development for *iALC*.

### 6.1

#### Reasoner Evaluation

For this activity, we used the approach shown in [78]. In this book, the authors define some phases of experimentation in software engineering:

- Scoping phase.
- Planning phase.
- Operation phase.
- Analysis and Interpretation phase.

<sup>1</sup>Competency Questions are used in methodologies for validating ontology functional requirements [75].

<sup>2</sup>A computational framework to validate competence questions by automatic reasoning on the SUMO ontology [76]

### 6.1.1 Scoping

For this case study, the Goal Question Metric paradigm (GQM) was adopted. The GQM is a paradigm that defines a software quality measurement model based on three levels: the conceptual level (Goal), the operational level (Question), and the quantitative level (Metric) [79].

In the next three subsections we formulate the objective of the case study, the questions to be answered (CQs), and the essential measures to produce the answers. The three aspects of evaluation are described below.

**Goal:**

Evaluate the performance of the KRM SAT solver in solving intuitionistic logic (*iALC*) problems in terms of accuracy and query coverage.

**Questions:**

$Q_1$  = Does KRM perform CQs validations by showing counter-examples through our tableau system's (automatic reasoning) on logical formulas in *iALC*?

$Q_2$  = Does KRM perform CQs validations into the relative number of SPARQL queries generated by counter-models?

**Metrics:** The engineering process uses some observable characteristics to define software quality. Some examples are cited in [79]: time, number of defects, defect severity, complexity, lines of code, coding effort, and productivity.

For this work, we defined two metrics ( $M_1$ ,  $M_2$ ):

$M_1$ . *Accuracy*: it is the measure of dividing the number of counter-examples (complement of *iALC* models) correctly inferred by the total number of CQs.

$M_2$ . *Query coverage*: it is the measure of dividing the number of counter-examples (defined by specialists) by the total number of queries generated correctly by KRM. Many times we will have a number of counter-models greater than the number of SPARQL queries. This is justified because we decided on just one more complete counter-model (greater number of terms among the counter-models) for its transformation into a query.

### 6.1.2 Planning

The planning prepares for *how* the experiment is conducted. For it, the planning phase of an experiment can be divided into seven steps:

- **Context Selection:** In this phase, we select the environment in which the experiment will be executed. The objective of the experiment is to

analyze the technical feasibility of our KRM SAT Solver. To achieve this, we defined a set of CQs extracted from various laws (municipal, state, and federal) together with business specialists from the city of Maceió. These CQs are formulas in *iALC* that represent the legislation’s compliance rules over *real problems*. Basically, these rules define what should be valid in a knowledge base in terms of its TBox, the domain of interpretation.

To run the experiment, examples of the  $RDF_{og}$  knowledge base were built to execute each competency question. The complexity and diversity of the laws addressed in the competency questions required a detailed and systematic analysis strategy, which can be facilitated by the availability of small concrete examples of official gazette data. In practice, for each CQ, examples of RDF instances were created with information in compliance and information about non-compliance with the legislation.

Furthermore, by using examples of compliance and non-compliance in the RDF, the experimentation process of this research can ensure a more objective and consistent evaluation of our solution responses since the evaluation criteria can be aligned with the data available in the examples. In this way, the construction of these examples, aligned with the CQs, not only facilitates understanding and analysis but also facilitates a more effective evaluation of our solution.

An important point, when defining the scope of this research, our tool does not perform a conjunctive ABox query on the RDF graph. In the Listing 14 is an example of a SPARQL query that makes a query ABox conjunctiva in specific class instances where um Person is filtered with age  $\geq 18$ .

---

**Listing 14** Conjunctive ABox query.

---

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ex: <http://example.org/>

SELECT ?subject
WHERE {
  ?subject rdf:type ex:Person .
  ?subject ex:hasAge ?age .
  FILTER (?age >= 18)
}
```

---

This query would return the subjects (instances) that would meet the specified criteria.

- **Hypothesis Formulation:** A hypothesis is formally stated, and the data collected during the experiment is utilized, if possible, to reject the hypothesis. If the hypothesis can be rejected, conclusions can then be drawn based on hypothesis testing [78]. The *Null Hypothesis* ( $H_0$ ), is the hypothesis that the experimenter wants to reject with as high significance as possible. The *Alternative Hypothesis* ( $H_1$ ), is a proposition that contrasts with the null hypothesis.

We defined the following statistical values  $\mu$  (populational average) for *Null Hypothesis* and *Alternative Hypothesis*:

#### Hypothesis Testing for Accuracy

- $H_0 : Accuracy < 1$
- $H_1 : Accuracy = 1$

#### Hypothesis Testing for Query Coverage

- $H_0 : QueryCoverage < 1$
- $H_1 : QueryCoverage = 1$

- **Variables Selection**

**Independent Variables:** In a hypothesis test, the independent variable is the one that you experimentally manipulate or control, which can affect the dependent variable, which is the one you are trying to measure or evaluate (in this case, the accuracy and query coverage metrics).

Therefore, the independent variable in this experiment refers to the 21 competency questions to be analyzed individually in the KRM SAT Solver.

#### Dependents Variable:

In a hypothesis test, the dependent variable is the one you are trying to understand or explain in relation to variations in the independent variables.

*Accuracy:* Accuracy is a performance measure that indicates the proportion of correctly inferred counter-examples to the total number of inferred counter-examples.

*Query Coverage:* Query coverage is another performance measure that indicates the proportion of counterexamples (generated by CQs) to the total number of queries generated by KRM.

- **Selection of Competency Questions as Study Subjects:** The appropriate selection of Competency Questions (CQs) is essential to ensure the validity and relevance of a study. In this section, we will describe the selection process of CQs written by experts based on different laws, who will serve as the subjects of study in our research.

*Definition of CQs:* Initially, we carefully reviewed the 20 expert-designed CQs, which were formulated based on various laws relevant to the public service domain. Our objective was to clearly identify and describe the issues addressed by each CQ and assess their relevance to the objectives of this research. The selected CQs must be able to provide valuable insights on how to identify nonconformities in the RDF database of public acts.

### 6.1.3 Operation

In this section, we detail the procedures used to carry out the proposed experiment. Remember that the objective of this experiment is to evaluate the performance of the KRM SAT solver by determining whether a given competency question (in *iALC* logic and respective counter-model) applied to this solver produces the same reference counter-model. Secondly, whether the SPARQL query is retrieving non-conformity information, in coherence with the counter-model. These evaluations will be in terms of *accuracy* and *query coverage*. For the reference counterexamples present in the CQs, manual proofs (*iALC tableau*) were carried out with the help of two students from the higher education course at IFAL (Instituto Federal de Alagoas). This evidence is stored in the experiment report.

#### ***Execution:***

Thus, the 21 CQs presented in Appendix A of this thesis were applied individually to this experiment as input to the KRM SAT solver. After execution, the counter-examples and SPARQL queries generated were stored in a log file for this experiment. Furthermore, each query was applied to AllegroGraph, which stores some counter-model examples in RDF format. Then, the result of this consultation was carefully recorded.

In order not to consume space in this thesis document, we will demonstrate this entire process of executing the competency questions in our tool, generating the proof trees and all the models and SPARQL query results, only for competence question numbers 1, 4, and 5. However, the result of the complete execution of this experiment was recorded and available in the **Ex-**



periment Report Link<sup>3</sup>.

Thus, in Figure 6.1, we present the sample of the solution tree for CQ number 1. The Listing 15 shows the Log file with the generated counter-model and Listing 16, the Log file with the generated SPARQL query. Remember that KRM SAT Solver is also capable of generating a log file with the model, when it is the case.

### Case study: $CQ_1$

Let us start our experiment with  $CQ_1$ . Law N<sup>o</sup> 7.713 of December 22, 1988, it establishes that retired employees and employees with severe diseases are exempt from income tax deductions [80]. Then, model this legal rule with the following specification in *iALC*:

$$\vdash_{iALC} (w : (\forall \text{tipoFunc.APOSENT} \sqcap \forall \text{tipoDoenca.INCAPAC}) \multimap \neg(\exists \text{temDesconto.IRPF})) \multimap \neg(\exists \text{temDesconto.IRPF})$$

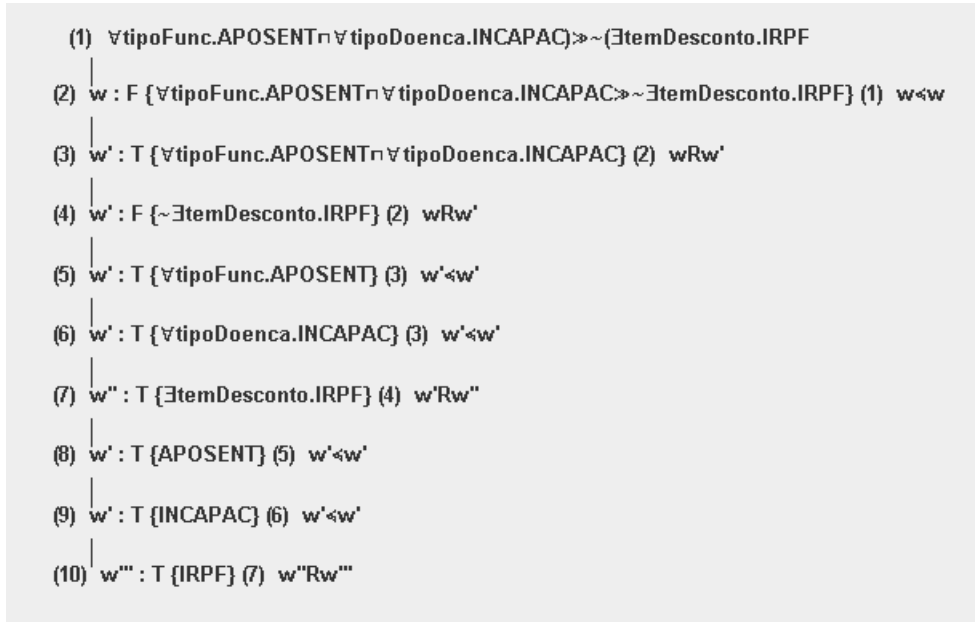


Figure 6.1: Proof tree of  $CQ_1$ .

As we can see, this proof tree presents on node 1 the original formula of our modeling. The *iALC* tableau calculation process begins by applying the first rule: the negation of the formula (Id node 2). Remembering that this tree has, in addition to the subformulas derived from the application of our calculation rules (Section 5.3.2), some meta-information. On the right side, there is the index of the original formula of this current subformula,

<sup>3</sup>[https://drive.google.com/file/d/1JdjhgctuAcuaq8EdS3wPibz-V1rMzcjX/view?usp=drive\\_link](https://drive.google.com/file/d/1JdjhgctuAcuaq8EdS3wPibz-V1rMzcjX/view?usp=drive_link)

in addition to information on the order between worlds and the relationship between worlds. This tableau, which is not closed, has a counter-example with three atomic subformulas (Listing 15). The  $wRw'$ ,  $w'Rw''$  and  $w''Rw'''$  relations guarantee the relationship between the worlds of these formulas. In practice, these relationships guarantee the existence of accessibility permission between these legal individuals (represented by these worlds).

---

**Listing 15** Counter-example Log generated to  $CQ_1$  by KRM SAT solve.

---

(10) w' ' ' T IRPF    (9) w' T INCAPAC    (8) w' T APOSENT

---

The Listing 15 shows three atomics nominal which represents a generated counter-example. This instance is then used as configuration parameters for a SPARQL query. By relating the detection of counter-examples through tableau with the parameters used in the SPARQL query (Listing 16), it is possible to establish a comprehensive approach to identify and correct errors in the knowledge base, as we will see below.

---

**Listing 16** SPARQL query generated to  $CQ_1$  by KRM SAT Solver.

---

```
PREFIX foaf:      <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
  {
    SELECT ?name
    WHERE {
      ?x foaf:temDesconto  'IRPF' .
      ?x foaf:name ?name .
    }
  }
  {
    SELECT ?name
    WHERE {
      ?x foaf:tipoDoenca  'INCAPAC' .
      ?x foaf:name ?name .
    }
  }
  {
    SELECT ?name
    WHERE {
      ?x foaf:tipoFunc  'APOSENT' .
      ?x foaf:name ?name .
    }
  }
}
```

---

In Listing 17 we can see that the RDF of experiment  $CQ_1$  contains four instances that represent the IRPF discount rule for four employees. Among these, two are in a situation of non-compliance with the law that defines the rule for exemption from the IRPF discount. The law, which can be seen in Appendix A, says that the non-deduction situation is for cases where the employee is retired and has a serious disease.

**Listing 17** RDF to  $CQ_1$ .

---

```

<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

  <foaf:Faixas rdf:nodeID="Nf1002b3c57c841a9bc83f3225c88f6cd">
    <foaf:tipoFunc>APOSENT</foaf:tipoFunc>
    <foaf:name>MARCELO SOUTO AIRES</foaf:name>
    <foaf:tipoDoenca>INCAPAC</foaf:tipoDoenca>
    <foaf:temDesconto>IRPF</foaf:temDesconto>
  </foaf:Faixas>
  <foaf:Faixas rdf:nodeID="Nf1002b3c57c841a9bc83f3225c88f6c5">
    <foaf:tipoFunc>APOSENT</foaf:tipoFunc>
    <foaf:name>EDUARDO LIMA NOVAES</foaf:name>
    <foaf:tipoDoenca>INCAPAC</foaf:tipoDoenca>
    <foaf:temDesconto>IRPF</foaf:temDesconto>
  </foaf:Faixas>
  <foaf:Faixas rdf:nodeID="N22e140cbdf804107b26cb49b537834b1">
    <foaf:tipoFunc>ATIVO</foaf:tipoFunc>
    <foaf:name>FERNANDO ANTONIO PINTO</foaf:name>
    <foaf:tipoDoenca>INCAPAC</foaf:tipoDoenca>
    <foaf:temDesconto>IRPF</foaf:temDesconto>
  </foaf:Faixas>
  <foaf:Faixas rdf:nodeID="N6388d7508deb49e585c9b699d9ad03b8">
    <foaf:tipoFunc>APOSENT</foaf:tipoFunc>
    <foaf:name>JOAO NOGUEIRA DA SILVA</foaf:name>
    <foaf:tipoDoenca>COMUM</foaf:tipoDoenca>
    <foaf:temDesconto>IRPF</foaf:temDesconto>
  </foaf:Faixas>
</rdf:RDF>

```

---

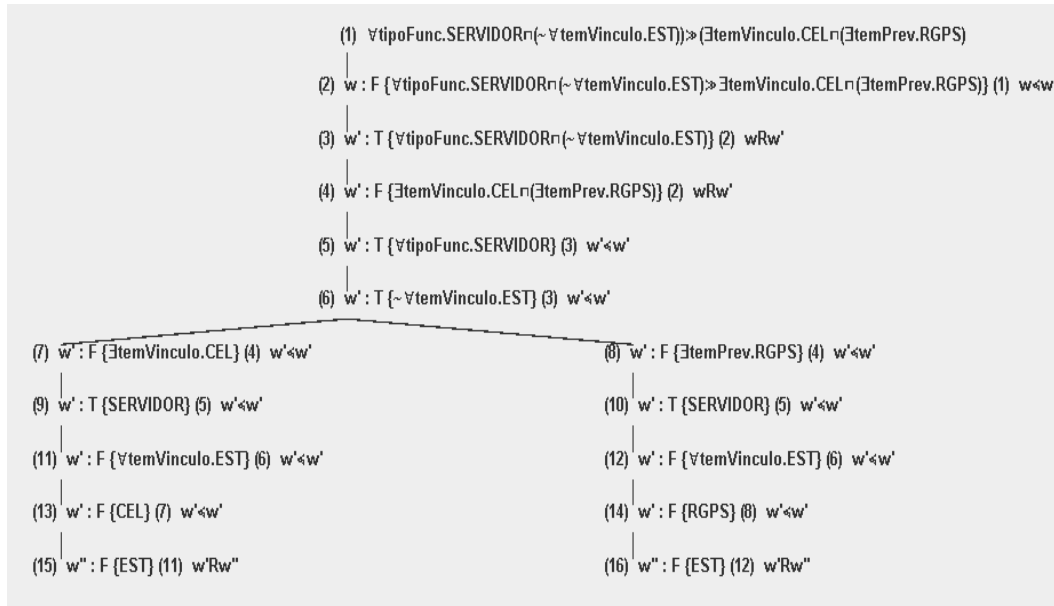
The result of executing the query, Figure 6.2, proves that our query is covering the situations imposed in the modeling of the *iALC* formula (If the employee is retired and has a serious disease, then cannot deduct income tax). The first case (in the result set) is Marcelo Souto Aires, who is retired (<foaf:tipoFunc>APOSENT), has a severe disease (<foaf:tipoDoenca>INCAPAC) and deducts it from the IRPF (<foaf:temDesconto>IRPF). The second case, Eduardo Lima Novaes, is similar to the first case.

2 ROWS	DOWNLOAD RESULTS	QUERY INFORMATION
name		
"EDUARDO LIMA NOVAES"		
"MARCELO SOUTO AIRES"		

Figure 6.2: Result query generated to  $CQ_1$  by AllegroGraph.

Case study:  $CQ_4$ 

The second experimenter is the  $CQ_4$  interpretation. According to Law No. 8,647 of April 13, 1993. Provides for the connection of a civil public employee, occupying a position on a commission without an effective link with the Federal Public Administration, to the RGPS (General Social Security Regime) [81]. The second interpretation involves the law No. 9,717 of November 27, 1998. Provides general rules for the organization and operation of social security systems for public employees of the Union, States, the Federal District and municipality, military personnel of the States, and the Federal District and provides other measures. Art. 1 - Art. 1 The specific social security rules (RPPS) for public employees in the Union, the States, the Federal District, and municipality[...] must be organized[...] [82].

$$(\forall \text{tipoFunc.SERVIDOR} \sqcap \neg(\forall \text{temVinculo.EST})) \rightarrow (\exists \text{temVinculo.CEL} \sqcap (\exists \text{temPrev.RGPS}))$$
Figure 6.3: Proof tree of  $CQ_4$ .

The proof tree of  $CQ_4$  (Figure 6.3), not a closed tableau, has a counter-model with two counter-examples, represented for two branches. The left branch contains the counter-example of nodes 15, 13, and 9 IDs. The right, counter-example of nodes 16, 14, and 10.

---

**Listing 18** Counter-model Log generated to  $CQ_4$  by KRM SAT solver.

---

(15)	w'	F	EST	(13)	w'	F	CEL	(9)	w'	T	SERVIDOR
(16)	w'	F	EST	(14)	w'	F	RGPS	(10)	w'	T	SERVIDOR

---

In cases where there is more than one counter-example, our transpiler generates a SPARQL query for each counter-example. As we can see List 19 (left branch) and List 20 (right branch).

In cases where the counter-example contains the “F” (false signal) of a nominal (nodes 15, 13, 16, and 14), our transpiler decides a template with a non-existence filter. The `FILTER NOT EXISTS` allows to filter results based on whether certain triple patterns do not exist in an RDF graph [83]. This is useful for queries where we want to find entities that do not have certain relationships or properties in common with other entities.

---

**Listing 19** SPARQL query generated to  $CQ_4$  (left branch) by KRM SAT Solver.

---

```

PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
  {
    SELECT ?name
    WHERE {
      ?x foaf:tipoFunc 'SERVIDOR' .
      FILTER NOT EXISTS {?x foaf:temVinculo 'EST'}
      ?x foaf:name ?name .
    }
  } {
    SELECT ?name
    WHERE {
      ?x foaf:tipoFunc 'SERVIDOR' .
      FILTER NOT EXISTS {?x foaf:temVinculo 'CEL'}
      ?x foaf:name ?name .
    }
  }
}

```

---

Listing 21 shows the RDF file with four instances of functional relationships. The cases of non-compliance are MARCELO SOUTO AIRES and RONALDO NAZARIO DOS SANTOS where the values of the predicates for `<foaf:temVinculo>` and `<foaf:temPrev>` are not compatible. Furthermore, the formula is looking for cases where employee not is “EST” (Estatutario) and “RGPS”. Therefore, only “RONALDO NAZARIO DOS SANTOS” (Figure 6.4) is retrieved by the query. The Experimenter  $CQ_5$  will show this complement.

1 ROW	DOWNLOAD RESULTS	QUERY INFORMATION
name		
"RONALDO NAZARIO DOS SANTOS"		

Figure 6.4: Result query generated to  $CQ_4$  by AllegroGraph.

---

**Listing 20** SPARQL query generated to  $CQ_4$  (right branch) by KRM SAT Solver.

---

```

PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
  {
    SELECT ?name
    WHERE {
      ?x foaf:tipoFunc 'SERVIDOR' .
      FILTER NOT EXISTS {?x foaf:temVinculo 'EST'}
      ?x foaf:name ?name .
    }
  } {
    SELECT ?name
    WHERE {
      ?x foaf:tipoFunc 'SERVIDOR' .
      FILTER NOT EXISTS {?x foaf:temPrev 'RGPS'}
      ?x foaf:name ?name .
    }
  }
}

```

---



---

**Listing 21** RDF to  $CQ_4$ .

---

```

<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

  <foaf:Vinculos rdf:nodeID="Nf1002b3c57c841a9bc83f3225c88f6XX">
    <foaf:name>MARCELO SOUTO AIRES</foaf:name>
    <foaf:tipoFunc>SERVIDOR</foaf:tipoFunc>
    <foaf:temVinculo>EST</foaf:temVinculo>
    <foaf:temPrev>RGPS</foaf:temPrev>
  </foaf:Vinculos>

  <foaf:Vinculos rdf:nodeID="N22e140cbdf804107b26cb49b537834YY">
    <foaf:name>FERNANDO ANTONIO DANTAS</foaf:name>
    <foaf:tipoFunc>SERVIDOR</foaf:tipoFunc>
    <foaf:temVinculo>EST</foaf:temVinculo>
    <foaf:temPrev>RPPS</foaf:temPrev>
  </foaf:Vinculos>

  <foaf:Vinculos rdf:nodeID="N6388d7508deb49e585c9b699d9ad03DD">
    <foaf:name>RONALDO NAZARIO DOS SANTOS</foaf:name>
    <foaf:tipoFunc>SERVIDOR</foaf:tipoFunc>
    <foaf:temVinculo>CEL</foaf:temVinculo>
    <foaf:temPrev>RPPS</foaf:temPrev>
  </foaf:Vinculos>
</rdf:RDF>

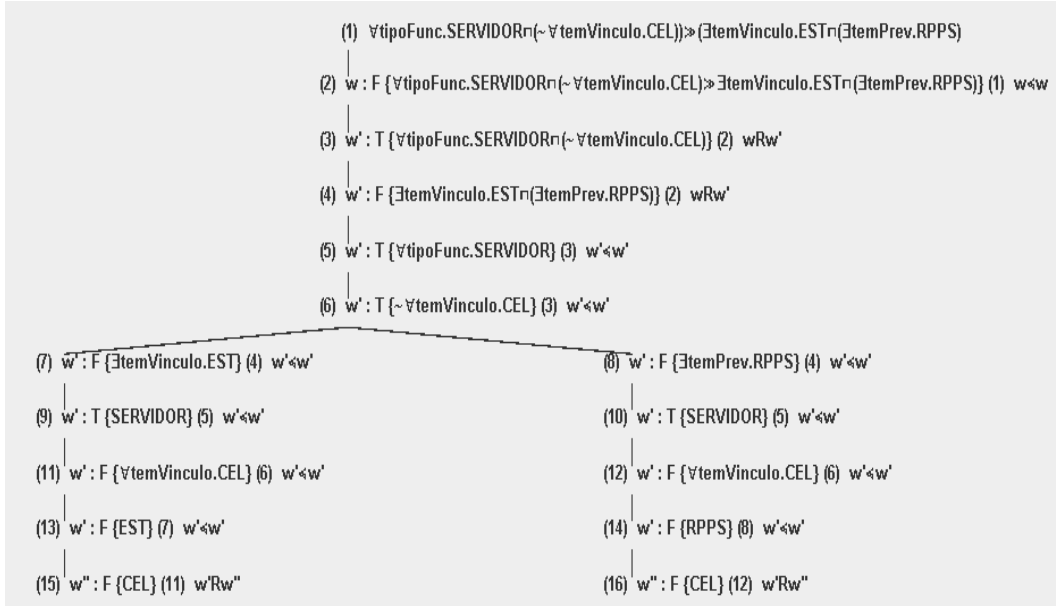
```

---

### *Case study: $CQ_5$*

This experiment is complementary to experiment  $CQ_4$ . Now we are interested in cases that do not meet this rule:

$$(\forall \text{tipoFunc.SERVIDOR} \sqcap \neg(\forall \text{temVinculo.CEL})) \rightarrow \\ (\exists \text{temVinculo.EST} \sqcap (\exists \text{temPrev.RPPS}))$$

Figure 6.5: Proof tree of  $CQ_5$ .

**Listing 22** Counter-model Log generated to  $CQ_5$  by KRM SAT solver.

(15)	$w'$	$F$	CEL	(13)	$w'$	$F$	EST	(9)	$w'$	$T$	SERVIDOR
(16)	$w'$	$F$	CEL	(14)	$w'$	$F$	RPPS	(10)	$w'$	$T$	SERVIDOR

Figure 6.6 shows the result of executing the SPARQL query on an example RDF base.

1 ROW	DOWNLOAD RESULTS	QUERY INFORMATION
name		
"MARCELO SOUTO AIRES"		

Figure 6.6: Result query generated to  $CQ_5$  by AllegroGraph.

**Data Collection:**

We present, in Table 6.1 below, the number of CQs validated according to the correctly generated counter-models.

Table 6.1: Summary of correct and incorrect CQs.

Quantity of CQs	Correct CQs	Incorrect CQs
21	21	0

Table 6.2 presents a summary of the results obtained from a close inspection of SPARQL queries in our  $RDF_{og}$  knowledge base. The main metric analyzed is the coverage of the SPARQL query, which indicates whether the query correctly returns the counter-model data.

The coverage of a SPARQL query is a crucial indicator of the effectiveness of our  $RDF_{og}$  KB and the accuracy of the queries performed. Successful coverage confirms that the queries return the expected results, while poor coverage may indicate flaws in the query logic or problems in the  $RDF_{og}$  structure.

To facilitate the traceability of this experiment, a set of RDF triples was constructed with fictitious information (compliance and non-compliance cases). 21 RDF bases were built, specialized for each legal norm. These RDFs are available in the experiment report.

Table 6.2: Query coverage summary.

CQ index	Correct coverage	Incorrect coverage	Observations
$CQ_{1-21}$	21	0	

This table is divided into four informative columns that provide a detailed view of the inspection results. The lines represent a summary of SPARQL queries, including information such as CQ index, correct coverage and incorrect coverage obtained, and additional observations (when applicable). This table provides a comprehensive view of the performance of our SPARQL queries with respect to data coverage. Based on this information, we can identify areas for improvement and implement strategies to optimize our SPARQL queries or improvements in the  $RDF_{og}$  KB.

**6.1.4****Analysis and Interpretation**

After running each CQ in our *SAT Solver*, the results were grouped to calculate the *accuracy* ( $M_1$ ) and the *query coverage* ( $M_2$ ) metrics of the CQ validations performed by KRM.

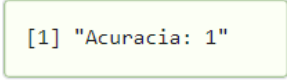


### Evaluating the SAT Solver generating correct counter-examples:

The table 6.3 indicates the accuracy of the SAT Solver. It is the number of counter-examples generated by the KRM divided by the number of reference counter-examples in each competency question (see Appendix B). In fact, we're going to be a little more precise. We will consider not only the correct counter-examples, but also the correct nominal contained in each counter-example. The accuracy calculation program can be seen in Appendix C. This R program calculates the difference between the reference answers (obtained in the experiment) and the answers of the KRM. The execution result can be seen in Figure 6.7

Table 6.3: KRM SAT Solver Accuracy.

$M_1$ - Accuracy	1.00
------------------	------



```
[1] "Acuracia: 1"
```

Figure 6.7: Accuracy calculated by the R program.

### Evaluating the KRM SAT solver generating queries that answering correctly:

Table 6.4 indicates the coverage of queries generated by KRM SAT solver. It is the number of correct information retrieved from the  $\text{RDF}_{og}$  KB divided by the number of non-conformity examples present in these knowledge bases. This verification and validation of query coverage can be seen in the experiment report.

Table 6.4: KRM Query Coverage.

$M_2$ - Query coverage	1.00
------------------------	------

This work presented a logical approach to verify compliance with public acts. To achieve this, an architecture capable of carrying out all the operations necessary for a good audit was developed. It starts with a mechanism for extracting information contained in the official gazette and organizing it in the format of RDF graph nodes. Next, a proposed Knowledge Representation Base construction mechanism was proposed using the diary grammar and abstract production rules for this construction. And finally, a module that checks whether the public acts present in our knowledge base comply with legal standards.

To demonstrate the execution capacity of our approach, we carried out an experiment with some competence questions (KRB examples) modeled by experts in the financial, environmental, and public human resources areas. In the end, we verified that the accuracy of our proposal was 100% for the metric of a number of examples correctly inferred by KRM and 100% coverage for the metric of queries generated by KRM.

As noted, the KRB construction module still faces some challenges, given the very nature of the problem of formalizing and extracting information from sources in context-sensitive grammar. We realized that when the grammar of the official diary is well-defined, this extraction process becomes viable.

In conclusion, the use of semantic tableau to analyze formulas in legal logic proved to be an interesting tool in detecting non-compliance in public knowledge bases. As we have seen, a counter-example generated by a semantic tableau reveals an instance in which the logical rules are not satisfied, thus highlighting possible scenarios of failures in the application of legislation. By relating the detection of counter-examples through semantic tables with the parameters used in SPARQL queries, it is possible to establish a comprehensive approach to identify and correct failures, thus contributing to ensuring the integrity and consistency of information.

For future work, we can list some work necessary for the evolution of this approach:

- The construction of a plugin for the Eclipse IDE [84], for example, where the user could carry out this entire process with the help of graphic

resources that every IDE presents.

- Development of the Formula Generator Module (FGM). It involves a practical way of reading the law and implementing its attribute grammar.
- *iALC* tableau improvements with known optimization techniques found in the Descriptive Logic literature (normalization, simplification, absorption, semantic branching, Backjumping, Caching Satisfiability Status, Top-Bottom Search for Classification and Model Merging) [85] and [11].
- A study and implementation of new templates for more complex SPARQL queries.
- A helper for constructing *iALC* formulas for cases where their construction by FGM is impossible. This builder would be part of the plugin features in Eclipse.
- Application our architecture for other data models and considering other theories and knowledge bases that could allow other interesting studies inferences.

## Bibliography

- [1] Brasil, Emenda Constitucional nº 9, de 9 de novembro de 1995, Diário Oficial [da] República Federativa do Brasil 59 (1995) 1966.
- [2] Brasil, Lei nº. 12.527. Lei de Acesso à Informação., Diário Oficial [da] República Federativa do Brasil (2011).
- [3] Brasil, Lei complementar nº 95, de 26 de fevereiro de 1998, Diário Oficial [da] República Federativa do Brasil (1998).  
URL [http://www.planalto.gov.br/ccivil\\_03/leis/lcp/lcp95.htm](http://www.planalto.gov.br/ccivil_03/leis/lcp/lcp95.htm)
- [4] Brasil, Lei complementar nº 107, de 26 de abril de 2001, Diário Oficial [da] República Federativa do Brasil (2001).  
URL [http://www.planalto.gov.br/ccivil\\_03/leis/lcp/lcp107.htm](http://www.planalto.gov.br/ccivil_03/leis/lcp/lcp107.htm)
- [5] H. Kelsen, Teoria pura do direito, 8th Edition, WMF Martins Fontes, São Paulo, 2009, iISBN: 83-336-0836-5.
- [6] E. H. Haeusler, A. Rademaker, On How Kelsenian Jurisprudence and Intuitionistic Logic help to avoid Contrary-to-Duty paradoxes in Legal Ontologies, in: Lógica no Avião Seminars, Vol. 1, Univeristy of Brasília, 2019, pp. 44–59.  
URL <http://doi.org/c768>
- [7] E. H. Haeusler, V. De Paiva, A. Rademaker, Intuitionistic Logic and Legal Ontologies, in: Legal Knowledge and Information Systems - JURIX 2010, Vol. 223 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2010, pp. 155–158.  
URL <https://doi.org/10.3233/978-1-60750-682-9-155>
- [8] J. A. Goguen, R. M. Burstall, Introducing institutions, in: E. Clarke, D. Kozen (Eds.), Logics of Programs, Springer Berlin Heidelberg, Berlin, Heidelberg, 1984, pp. 221–256.
- [9] K. J. Barwise, Axioms for abstract model theory, Annals of Mathematical Logic 7 (2-3) (1974) 221–265. doi:10.1016/0003-4843(74)90016-3.

- [10] A. Martini, U. Wolter, E. H. Haeusler, Fibred and Indexed Categories for Abstract Model Theory, *Logic Journal of the IGPL* 15 (5-6) (2007) 707–739. doi:10.1093/jigpal/jzm045.  
URL <https://doi.org/10.1093/jigpal/jzm045>
- [11] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation, and Applications*. ISBN: 978-0-521-15011-8, Cambridge University Press; 2nd Edition, 2007.
- [12] F. A. Pinto, E. Haeusler, S. Lifschitz, Transparência pública automatizada a partir da gramática do diário oficial, in: *Anais do IX Workshop de Computação Aplicada em Governo Eletrônico*, SBC, Porto Alegre, RS, Brasil, 2021, pp. 59–70. doi:10.5753/wcge.2021.15977.  
URL <https://sol.sbc.org.br/index.php/wcge/article/view/15977>
- [13] F. A. D. G. Pinto, S. Lifschitz, E. H. Haeusler, A knowledge base of public acts based on the grammar of the official gazette, in: *2022 International Conference on Digital Government Technology and Innovation (DGTi-CON)*, 2022, pp. 24–29. doi:10.1109/DGTi-CON53875.2022.9849196.
- [14] F. Pinto, S. Lifschitz, E. Haeusler, A graph knowledge-base for auditing human resources public management, in: *Anais do X Workshop de Computação Aplicada em Governo Eletrônico*, SBC, Porto Alegre, RS, Brasil, 2022, pp. 61–72. doi:10.5753/wcge.2022.223273.  
URL <https://sol.sbc.org.br/index.php/wcge/article/view/20711>
- [15] F. Pinto, J. Santos, S. Lifschitz, E. Haeusler, A benchmarking for public information by machine learning and regular language, in: *Anais do XI Workshop de Computação Aplicada em Governo Eletrônico*, SBC, Porto Alegre, RS, Brasil, 2023, pp. 60–71. doi:10.5753/wcge.2023.229975.  
URL <https://sol.sbc.org.br/index.php/wcge/article/view/24865>
- [16] IBM, IBM Watson Discovery, <https://www.ibm.com/br-pt/products/watson-discovery>, [Online; accessed 2023-03-03] (2023).
- [17] H. Prakken, G. Sartor, Law and logic: A review from an argumentation perspective, *Artificial Intelligence* 227 (2015) 214–245.

- doi:10.1016/j.artint.2015.06.005.  
URL <https://linkinghub.elsevier.com/retrieve/pii/S0004370215000910>
- [18] R. Pressman, B. Maxim, Engenharia De Software: UMA ABORDAGEM PROFISSIONAL, MCGRAW HILL - ARTMED, 2016.
- [19] R. Brachman, H. Levesque, Knowledge Representation and Reasoning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.
- [20] H. J. Levesque, Knowledge representation and reasoning, Annual Review of Computer Science 1 (1) (1986) 255–287. arXiv: <https://doi.org/10.1146/annurev.cs.01.060186.001351>, doi:10.1146/annurev.cs.01.060186.001351.  
URL <https://doi.org/10.1146/annurev.cs.01.060186.001351>
- [21] K. Constantino, V. A. L. Cruz, O. M. M. Zucheratto, C. França, M. Carvalho, T. H. P. Silva, A. H. F. Laender, M. A. Gonçalves, Segmentação e classificação semântica de trechos de diários oficiais usando aprendizado ativo, in: Anais do XXXVII Simpósio Brasileiro de Banco de Dados (SBBD 2022), Sociedade Brasileira de Computação - SBC, 2022, pp. 304–316. doi:10.5753/sbbd.2022.224656.  
URL <https://sol.sbc.org.br/index.php/sbbd/article/view/21815>
- [22] A. V. Aho, M. S. Lam, R. Sethi, J. D. Ullman, Compilers: Principles, Techniques, and Tools (2nd Edition), Addison-Wesley Longman Publishing Co., Inc., USA, 2006.
- [23] Rodríguez, M., Dantas Bezerra, B, Processamento de linguagem natural para reconhecimento de entidades nomeadas em textos jurídicos de atos administrativos (portarias), Revista de Engenharia e Pesquisa Aplicada 5 (1) (2019) 67–77.
- [24] C. Friedman, T. C. Rindflesch, M. Corn, Natural language processing: State of the art and prospects for significant progress, a workshop sponsored by the national library of medicine, Journal of Biomedical Informatics 46 (5) (2013) 765–773. doi:<https://doi.org/10.1016/j.jbi.2013.06.004>.  
URL <https://www.sciencedirect.com/science/article/pii/S1532046413000798>

- [25] E. Loper, S. Bird, Nltk: The natural language toolkit, in: In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics, 2002, p. 8.
- [26] R. Junior, W. Melo, R. Fagundes, A. Maciel, Extração de informação e mineração de dados no diário oficial de pernambuco, *Revista de Engenharia e Pesquisa Aplicada* 3 (08 2018). doi:10.25286/repa.v3i3.892.
- [27] C. Buil-Aranda, A. Hogan, J. Umbrich, P.-Y. Vandenbussche, Sparql web-querying infrastructure: Ready for action?, in: H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. X. Parreira, L. Aroyo, N. Noy, C. Welty, K. Janowicz (Eds.), *The Semantic Web – ISWC 2013*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 277–293.
- [28] L. Chiticariu, Y. Li, F. R. Reiss, Rule-based information extraction is dead! long live rule-based information extraction systems!, in: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Seattle, Washington, USA, 2013, pp. 827–832.  
URL <https://aclanthology.org/D13-1079>
- [29] B. Ford, Parsing expression grammars: a recognition-based syntactic foundation, *SIGPLAN Not.* 39 (1) (2004) 111–122. doi:10.1145/982962.964011.  
URL <https://doi.org/10.1145/982962.964011>
- [30] D. Beckett, Rdf/xml syntax specification (revised) w3c recommendation 10 february 2004, <http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/> (2007).
- [31] R. Team, RDFLib, <https://rdflib.readthedocs.io/en/stable/#>, acessado: 13-11-2019 (2013).  
URL <https://rdflib.readthedocs.io/en/stable/#>
- [32] D. Brickley, L. Miller, FOAF Vocabulary Specification 0.99, <http://www.foaf-project.org/>, acessado: 03-12-2019 (2014).  
URL <http://www.foaf-project.org/>
- [33] AllegroGraph, Allegrograph - The Enterprise Knowledge Graph, <https://allegrograph.com>, acessado: 11-12-2019 (2019).

- [34] D. E. Knuth, Semantics of context-free languages, *Mathematical systems theory* 2 (1968) 127–145.  
URL <https://link.springer.com/article/10.1007/BF01692511>
- [35] K. Koskimies, K.-J. Räihä, M. Sarjakoski, Compiler construction using attribute grammars, in: *Proceedings of the 1982 SIGPLAN Symposium on Compiler Construction, SIGPLAN '82*, Association for Computing Machinery, New York, NY, USA, 1982, p. 153–159. doi:10.1145/800230.806991.  
URL <https://doi.org/10.1145/800230.806991>
- [36] A. Aho, R. Sethi, M. Lam, J. Ullman, *Compiladores: Princípios, Técnicas e Ferramentas*, Pearson Universidades, 2007.
- [37] H.-L. Trieu, A.-K. Duong Nguyen, N. Nguyen, M. Miwa, H. Takamura, S. Ananiadou, Coreference resolution in full text articles with BERT and syntax-based mention filtering, in: K. Jin-Dong, N. Claire, B. Robert, D. Louise (Eds.), *Proceedings of the 5th Workshop on BioNLP Open Shared Tasks*, Association for Computational Linguistics, Hong Kong, China, 2019, pp. 196–205. doi:10.18653/v1/D19-5727.  
URL <https://aclanthology.org/D19-5727>
- [38] B. Z. Li, G. Stanovsky, L. Zettlemoyer, Active learning for coreference resolution using discrete annotation (2020). arXiv:2004.13671.
- [39] R. Vieira, A. Mendes, P. Quaresma, E. B. da Fonseca, S. Collovini, S. Antunes, Corref-pt: A semi-automatic annotated portuguese coreference corpus, *Computación y Sistemas* 22 (2018).  
URL <https://api.semanticscholar.org/CorpusID:59523789>
- [40] E. Fonseca, A. Antonitsch, S. Collovini, D. Amaral, R. Vieira, A. Figueira, Summ-it++: an enriched version of the summ-it corpus, in: N. Calzolari, K. Choukri, T. Declerck, S. Goggi, M. Grobelnik, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis (Eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, European Language Resources Association (ELRA), Portorož, Slovenia, 2016, pp. 2047–2051.  
URL <https://aclanthology.org/L16-1324>



- [41] S. Group, Industrial-strength. natural language processing, <https://spacy.io/>, accessed: 06-06-2021 (2020).  
URL <https://spacy.io/>
- [42] A. Tsvetkova, Anaphora resolution in chinese for analysis of medical q&a platforms, in: Natural Language Processing and Chinese Computing: 9th CCF International Conference, NLPCC 2020, Zhengzhou, China, October 14–18, 2020, Proceedings, Part II 9, Springer, 2020, pp. 490–497.
- [43] M. C. Fitting, Intuitionistic logic model theory and forcing, North-Holland Publishing, Amsterdam, Netherlands, 1969.
- [44] F. Baader, I. Horrocks, U. Sattler, Chapter 3 description logics, in: F. van Harmelen, V. Lifschitz, B. Porter (Eds.), Handbook of Knowledge Representation, Vol. 3 of Foundations of Artificial Intelligence, Elsevier, 2008, pp. 135–179. doi:[https://doi.org/10.1016/S1574-6526\(07\)03003-9](https://doi.org/10.1016/S1574-6526(07)03003-9).  
URL <https://www.sciencedirect.com/science/article/pii/S1574652607030039>
- [45] W3C, OWL web ontology language reference, <https://www.w3.org/TR/owl-ref/>, accessed: 09/11/2023 (2004).  
URL <https://www.w3.org/TR/owl-ref/>
- [46] G. De Giacomo, M. Lenzerini, et al., Tbox and abox reasoning in expressive description logics., KR 96 (316–327) (1996) 10.
- [47] A. Fokoue, A. Kershenbaum, L. Ma, E. Schonberg, K. Srinivas, The summary abox: Cutting ontologies down to size, in: I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, L. M. Aroyo (Eds.), The Semantic Web - ISWC 2006, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 343–356.
- [48] B. Alkmim, E. H. Haeusler, A. Rademaker, Utilizing ialc to formalize the brazilian OAB exam, in: G. J. Nalepa, M. Atzmueller, M. Araszkievicz, P. Novais (Eds.), Proceedings of the EXplainable AI in Law Workshop co-located with the 31st International Conference on Legal Knowledge and Information Systems, XAILA@JURIX 2018, Groningen, The Netherlands, December 12, 2018, Vol. 2381 of CEUR Workshop Proceedings, CEUR-WS.org, 2018, pp. 42–50.  
URL [https://ceur-ws.org/Vol-2381/xaila2018\\_paper\\_2.pdf](https://ceur-ws.org/Vol-2381/xaila2018_paper_2.pdf)

- [49] P. Blackburn, M. de Rijke, Y. Venema, *Modal Logic*, no. 53 in *Cambridge Tracts in Theoretical Computer Science*, Cambridge University Press, Cambridge, UK, 2001.
- [50] E. H. Haeusler, V. d. Paiva, A. Rademaker, Intuitionistic description logic and legal reasoning, in: *2011 22nd International Workshop on Database and Expert Systems Applications*, 2011, pp. 345–349. doi:10.1109/DEXA.2011.46.
- [51] H. Kelsen, *Pure Theory of Law*. ISBN: 9781584775782, Lawbook Exchange, 2005.
- [52] G. D. Plotkin, C. Stirling, A framework for intuitionistic modal logics, in: *Proceedings of the 1st Conference on Theoretical Aspects of Reasoning about Knowledge*, Monterey, CA, USA, March 1986, 1986, pp. 399–406.
- [53] G. T. Bagni, The first century of the international comission on mathematical instruction (1908-2008), <https://www.icmihistory.unito.it/portrait/beth.php>, accessed: 22-12-2023 (2020).  
URL <https://www.icmihistory.unito.it/portrait/beth.php>
- [54] M. D’Agostino, D. M. Gabbay, R. Hahnle, J. Posegga, *Handbook of Tableau Methods*, Springer Dordrecht, 1999.
- [55] I. Anellis, From semantic tableaux to smullyan trees: A history of the development of the falsifiability tree method, *The Review of Modern Logic* 1 (06 1990).
- [56] S. Jaśkowski, On the Rules of Suppositions in Formal Logic, *Studia Logica* 1 (1934) 5–32, (reprinted in: Storrs McCall (ed.), *Polish Logic 1920–1939*, Oxford University Press, 1967 pp. 232–258).  
URL <https://www.logik.ch/daten/jaskowski.pdf>
- [57] G. Gentzen, Investigations into logical deduction, *American Philosophical Quarterly* 1 (4) (1964) 288–306.
- [58] A. K. Simpson, The proof theory and semantics of intuitionistic modal logic, Ph.D. thesis, University of Edinburgh, UK (1994).  
URL <http://hdl.handle.net/1842/407>
- [59] M. D. P. N. Medeiros, A New S4 Classical Modal Logic in Natural Deduction, *The Journal of Symbolic Logic* 71 (3) (2006) 799–809.  
URL <http://www.jstor.org/stable/27588483>

- [60] A. Rademaker, A Proof Theory for Description Logics, Springer Publishing Company, Incorporated, 2012.
- [61] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, Y. Katz, Pellet: A practical OWL-DL reasoner, *Journal of Web Semantics* 5 (2) (2007) 51–53. doi:<https://doi.org/10.1016/j.websem.2007.03.004>.  
URL <https://www.sciencedirect.com/science/article/pii/S1570826807000169>
- [62] D. Tsarkov, I. Horrocks, Fact++ description logic reasoner: system description, in: *Proceedings of the Third International Joint Conference on Automated Reasoning, IJCAR’06*, Springer-Verlag, Berlin, Heidelberg, 2006, p. 292–297. doi:[10.1007/11814771\\_26](https://doi.org/10.1007/11814771_26).  
URL [https://doi.org/10.1007/11814771\\_26](https://doi.org/10.1007/11814771_26)
- [63] J. Bondy, U. Murty, *Graph Theory*, 1st Edition, Springer Publishing Company, Incorporated, 2008.  
URL <https://dl.acm.org/doi/10.5555/1481153>
- [64] H.-D. A. Hiep, O. Maathuis, J. Bian, F. S. de Boer, M. van Eekelen, S. de Gouw, Verifying openjdk’s linkedlist using key (2019). arXiv:1911.04195.
- [65] T. Parr, *The Definitive ANTLR 4 Reference*, 2nd Edition, Pragmatic Bookshelf, Raleigh, NC, 2013.  
URL <https://www.safaribooksonline.com/library/view/the-definitive-antlr/9781941222621/>
- [66] S. Perugini, *Programming Languages: Concepts and Implementation*, Jones & Bartlett Learning, 2021.
- [67] S. Dasgupta, C. H. Papadimitriou, U. Vazirani, *Algorithms*, 1st Edition, McGraw-Hill, Inc., USA, 2006.
- [68] B. Motik, R. D. C. Shearer, I. Horrocks, Hypertableau reasoning for description logics, *J. Artif. Intell. Res.* 36 (2009) 165–228. doi:[10.1613/JAIR.2811](https://doi.org/10.1613/JAIR.2811).  
URL <https://doi.org/10.1613/jair.2811>
- [69] L. T. Keith D. Cooper, *Engineering a Compiler*, 2nd Edition, Elsevier, 2011.
- [70] S. Powers, *Practical RDF: Solving Problems with the Resource Description Framework*, O’Reilly, Beijing, 2003.

- URL <https://www.safaribooksonline.com/library/view/practical-rdf/0596002637/>
- [71] R. W. Group, Resource Description Framework (RDF), <https://www.w3.org/RDF/>, acessado: 11-10-2019 (2014).  
URL <https://www.w3.org/RDF/>
- [72] G. Barzdins, D. Gosko, P. F. Barzdins, U. Lavrinovics, G. Bernans, E. Celms, Rdf\* graph database as interlingua for the textworld challenge, in: 2019 IEEE Conference on Games (CoG), 2019, pp. 1–2. doi:10.1109/CIG.2019.8848012.
- [73] S. Isotani, I. Bittencourt, Dados Abertos Conectados: em Busca da Web do Conhecimento, Novatec, 2015. doi:10.13140/RG.2.1.4355.6329.
- [74] G. Kellogg, P.-A. Champin, O. Hartig, A. Seaborne, RDF 1.2 concepts and abstract syntax, W3C working draft, W3C, <https://www.w3.org/TR/2024/WD-rdf12-concepts-20240121/> (Jan. 2024).
- [75] M. Gruninger, Methodology for the design and evaluation of ontologies, in: International Joint Conference on Artificial Intelligence, 1995, p. 10.  
URL <https://api.semanticscholar.org/CorpusID:16641142>
- [76] D. E. Oliveira, Coreacq: um framework computacional para validar questões de competência por raciocínio automático sobre a ontologia sumo, Master's thesis, Universidade Federal de Pernambuco, Recife (2019).  
URL <https://repositorio.ufpe.br/handle/123456789/33693>
- [77] C. Bezerra, F. Freitas, F. Santana, Evaluating ontologies with competency questions, in: 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), Vol. 3, 2013, pp. 284–285. doi:10.1109/WI-IAT.2013.199.
- [78] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, A. Wessln, Experimentation in Software Engineering, Springer Publishing Company, Incorporated, 2012.
- [79] V. R. Basili, G. Caldiera, H. D. Rombach, The Goal Question Metric Approach (1994).

- URL <https://api.semanticscholar.org/CorpusID:13884048>
- [80] Brasil, Lei nº 7.713, de 22 de dezembro de 1988, Diário Oficial [da] República Federativa do Brasil (1988).  
URL [https://www.planalto.gov.br/ccivil\\_03/leis/17713compilada.htm](https://www.planalto.gov.br/ccivil_03/leis/17713compilada.htm)
- [81] Brasil, Lei nº 8.647, de 13 de abril de 1993, Diário Oficial [da] República Federativa do Brasil (1993).  
URL [https://www.planalto.gov.br/ccivil\\_03/leis/L8647.htm](https://www.planalto.gov.br/ccivil_03/leis/L8647.htm)
- [82] Brasil, Lei nº 9.717, de 27 novembro de 1998, Diário Oficial [da] República Federativa do Brasil (1998).  
URL [https://www.planalto.gov.br/ccivil\\_03/leis/19717.htm](https://www.planalto.gov.br/ccivil_03/leis/19717.htm)
- [83] B. DuCharme, Learning SPARQL: Querying and Updating with SPARQL 1.1, 2nd Edition, O'Reilly, Beijing, 2013.
- [84] Eclipse Foundation, PDE UI, <https://eclipse.dev/pde/pde-ui/>, [Online; accessed 2024-04-04] (2024).
- [85] U. Hustadt, R. A. Schmidt, Simplification and Backjumping in Modal Tableau, in: H. de Swart (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 187–201.
- [86] BRASIL, Constituição da República Federativa do Brasil de 1988, Brasília, DF: Presidência da República (1988, accessed on 09/07/2023).  
URL [https://www2.senado.leg.br/bdsf/bitstream/handle/id/518231/CF88\\_Livro\\_EC91\\_2016.pdf](https://www2.senado.leg.br/bdsf/bitstream/handle/id/518231/CF88_Livro_EC91_2016.pdf)
- [87] Brasil, Lei complementar nº 8112, de 11 de dezembro de 1990, Diário Oficial [da] República Federativa do Brasil (1990).  
URL [https://www.planalto.gov.br/ccivil\\_03/leis/18112cons.htm](https://www.planalto.gov.br/ccivil_03/leis/18112cons.htm)
- [88] Brasil, Lei nº 13.303, de 30 de junho de 2016, Diário Oficial [da] República Federativa do Brasil (2016).  
URL [https://www.planalto.gov.br/ccivil\\_03/\\_ato2015-2018/2016/lei/113303.htm](https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2016/lei/113303.htm)

- [89] Maceió, Lei nº 6.987, de 11 de maio de 2020 (2020).  
URL <https://www.maceio.al.leg.br/documentos/docs/doc.php?filepath=leis&id=6676>
- [90] Maceió, Lei nº 6.685, de 18 de agosto de 2017 (2017).  
URL <https://www.maceio.al.leg.br/documentos/docs/doc.php?filepath=leis&id=6394>
- [91] TCE-Paraná, Acórdão nº 3561/23 - vedação à percepção de função gratificada por ocupante de cargo comissionado (nov 2023).  
URL [https://www.mpc.pr.gov.br/wp-content/uploads/2023/11/CORNELIO-PROCOPIO-ACO-3561\\_23-STP.pdf](https://www.mpc.pr.gov.br/wp-content/uploads/2023/11/CORNELIO-PROCOPIO-ACO-3561_23-STP.pdf)

## 8

## Appendix

This appendix is divided into four parts. Part “A” contains the legally formulated Competence Questions. To put it differently, we cite the legal basis (the law) for each Competency Question. Furthermore, we present the modeling of this law in *iALC* and its respective expected counterexample. Remembering that each Competency Question was duly formulated by a specialist, and confirmed by the legislation described. Part “B” of this appendix contains the grammar of the *iALC* language in ANLTR4, as well as attribute labels that assist in the walking process through the parsing tree. In Appendix “C”, an *R program* that calculates the accuracy of our experiment. Finally, in Appendix “D”, shows the soundness proof of the algorithm that implements the calculi system of the *iALC* tableau, Listing (1).

### Appendix A

#### Competency Questions

(Lei N<sup>o</sup> 7.713 DE 22/12/1988).Modify the legislation about Income Tax (similar to Federal Income Tax in the USA) and takes other measures. Art. 1<sup>o</sup> Income and capital gains received from January 1, 1989, by individuals resident or domiciled in Brazil, will be taxed by income tax in accordance with current legislation.

XIV – Retirement benefits and those with active tuberculosis, mental illness, multiple sclerosis, malignant neoplasia, blindness, leprosy, irreversible and disabling paralysis, severe heart disease, Parkinson’s disease, advanced stages of Paget’s disease, radiation contamination, immunodeficiency syndrome acquired, based on the conclusion of specialized medicine, even if the disease was contracted after retirement[...] [80]

$$\vdash_{iALC} (w : (\forall tipoFunc.APOSENT \sqcap \forall tipoDoenca.INCAPAC) \rightarrow \neg(\forall temDesconto.IRPF))$$

CQ<sub>1</sub> - Counterexample:

$w''' : \mathcal{T}$  IRPF ;  $w' : \mathcal{T}$  INCAPC ;  $w' : \mathcal{T}$  APOSENT

(Lei N<sup>o</sup> 8.647 DE 13/04/1993). Provides for the connection of a civil public employee, occupying a position on a commission without an effective link with the Federal Public Administration, to the RGPS (General Social Security Rule).

Art. 1<sup>o</sup> The civil public employee occupying a commission position, without an effective link with the Union, [...] is obligatorily linked to the RGPS (General Social Security Rule) referred to in Law N<sup>o</sup>. 8,213 of July 24, 1991 [81].

$\vdash_{iALC} (w : (\forall tipoFunc.SERVIDOR \sqcap \forall temCargo.COMISSIONADO) \multimap \neg(\exists temPrev.RPPS))$

CQ<sub>2</sub> - Counterexample:

$w' : \mathcal{T}$  SERVIDOR ;  $w' : \mathcal{T}$  COMISSIONADO ;  $w''' : \mathcal{T}$  RPPS

(Lei N<sup>o</sup> 9.717 DE 27/11/1998). Provides general rules for the organization and operation of social security systems for public employees of the Union, States, the Federal District and municipality, military personnel of the States, and the Federal District and provides other measures.

Art. 1 - Art. 1 The specific social security rules (RPPS) for public employees in the Union, the States, the Federal District, and municipality[...] must be organized[...] [82].

$\vdash_{iALC} (w : (\forall tipoFunc.SERVIDOR \sqcap \neg(\forall temVinculo.EST)) \multimap (\exists temPrev.RGPS))$

CQ<sub>3</sub> - Counterexample:

$w'' : \mathcal{F}$  EST ;  $w' : \mathcal{T}$  SERVIDOR ;  $w' : \mathcal{F}$  RGPS

$\vdash_{iALC} (w : (\forall tipoFunc.SERVIDOR \sqcap \neg(\forall temVinculo.EST)) \multimap (\exists temVinculo.CEL \sqcap \exists temPrev.RGPS))$

CQ<sub>4</sub> - Counterexample:

a)  $w'' : \mathcal{F}$  EST ;  $w' : \mathcal{F}$  CEL ;  $w' : \mathcal{T}$  SERVIDOR

b)  $w'' : \mathcal{F}$  EST ;  $w' : \mathcal{F}$  RGPS ;  $w' : \mathcal{T}$  SERVIDOR

$\vdash_{iALC} (w : (\forall tipoFunc.SERVIDOR \sqcap \neg(\forall temVinculo.CEL)) \multimap (\exists temVinculo.EST \sqcap \exists temPrev.RPPS))$

CQ<sub>5</sub> - Counterexample:

a)  $w'' : \mathcal{F}$  CEL ;  $w' : \mathcal{F}$  EST ;  $w' : \mathcal{T}$  SERVIDOR



b)  $w'' : \mathcal{F} \text{ CEL} ; w' : \mathcal{F} \text{ RPPS} ; w' : \mathcal{T} \text{ SERVIDOR}$

- Accumulation of public positions:

Constitution of the Federative Republic of Brazil of 1988. Constitutional Amendment N<sup>o</sup> 19, of 1998. [86]

XVI - the prohibited accumulation of public positions, except when there is compatibility of schedules, observing in any case the provisions of item XI:

a) Two teacher positions; b) A teacher position with another technical or scientific; c) Two positions or jobs exclusive to health professionals, with regulated professions;

$\vdash_{iALC} (w : (\forall temClasse.TECNICOA \sqcap \forall temParecer.ACUMULAR) \rightarrow (\forall temClasse.TECNICOA \sqcap \neg \forall temClasse.TECNICOB))$

CQ<sub>6</sub> - Counterexample:

$w'' : \mathcal{T} \text{ TECNICOB} ; w' : \mathcal{T} \text{ ACUMULAR} ; w' : \mathcal{T} \text{ TECNICOA}$

$\vdash_{iALC} (w : (\forall temClasse.TECNICOA \sqcap \forall temClasse.TECNICOB) \rightarrow \neg(\forall temParecer.ACUMULAR))$

CQ<sub>7</sub> - Counterexample:

$w'' : \mathcal{T} \text{ ACUMULAR} ; w' : \mathcal{T} \text{ TECNICOB} ; w' : \mathcal{T} \text{ TECNICOA}$

$\vdash_{iALC} (w : (\forall temClasse.MAGISTERIOA \sqcap \forall temClasse.TECNICOA) \rightarrow (\exists temParecer.ACUMULAR))$

CQ<sub>8</sub> - Counterexample:

$w' : \mathcal{T} \text{ TECNICOA} ; w' : \mathcal{T} \text{ MAGISTERIO} ; w' : \mathcal{F} \text{ ACUMULAR}$

- Become effective:

(Lei N<sup>o</sup> 8,112 DE 11/12/1990). Art. 21. The public employee qualified in a public examination and in a permanent position will acquire stability in the public service upon completing 2 (two) years of effective exercise [87].

$\vdash_{iALC} (w : (\forall frmIngresso.CONCURSADO \sqcap \forall temTempo.MENOR2ANOS) \rightarrow (\neg \exists temSituacao.EFETIVO))$

CQ<sub>9</sub> - Counterexample:

$w''' : \mathcal{T}$  EFETIVO ;  $w' : \mathcal{T}$  MENOR2ANOS ;  $w' : \mathcal{T}$  CONCURSADO

(Lei N<sup>o</sup> 13,303, of June 30, 2016). Provides for the legal status of public companies, within the scope of the Union, the States, the Federal District and the Municipalities.

Art. 17. The members of the Board of Directors and those nominated for the positions of director, including president, general director and chief executive officer, will be chosen from among citizens with an unblemished reputation and notable knowledge, and alternatively one of the requirements of paragraphs “a”, “b” and “c” of section I and, cumulatively, the requirements of sections II and III:

I - have professional experience of at least:

a) 10 (ten) years, in the public or private sector, in the area of activity of the public company or in an area related to that for which they are appointed as senior management; or

b) 4 (four) years in one of the following positions: [...] 3 - teaching or researcher position in areas of activity of the public company[...] [88].

$\vdash_{iALC} ((w : (\neg \forall temExper.DEZANOS) \sqcap (\neg \forall trabalhouComo.PESQUISADOR)) \rightarrow (\neg \exists staAtividade.ATIVO))$

CQ<sub>10</sub> - Counterexample:

$w'''' : \mathcal{F}$  PESQUISADOR ;  $w'''' : \mathcal{F}$  DEZANOS ;  $w''' : \mathcal{T}$  ATIVO

(Lei N<sup>o</sup> 13,303, of June 30, 2016). Provides for the legal status of public companies, within the scope of the Union, the States, the Federal District and the Municipalities.

Art. 22. The Board of Directors must be composed of at least 25% (twenty-five percent) of independent members[...], in accordance with Art. 141 of Law No. 6,404, of December 15, 1976.

III - not have maintained, in the last 3 (three) years, a relationship of any nature with the public company[...] [88].

$\vdash_{iALC} ((w : (\forall trabalhouOnde.EMPPUBLICA) \sqcap (\forall tempDecorrido.MENOS3ANOS)) \rightarrow (\neg \exists staAtividade.ATIVO))$

CQ<sub>11</sub> - Counterexample:

$w''' : \mathcal{T}$  ATIVO ;  $w' : \mathcal{T}$  MENOS3ANOS ;  $w' : \mathcal{T}$  EMPPUBLICA

$\vdash_{iALC} ((w : (\forall trabalhouOnde.EMPPUBLICA) \sqcap (\forall staAtividade.ATIVO)) \rightarrow (\neg \exists tempDecorrido.MENOS3ANOS))$

CQ<sub>12</sub> - Counterexample:

$w''' : \mathcal{T} \text{ MENOS3ANOS} ; w' : \mathcal{T} \text{ ATIVO} ; w' : \mathcal{T} \text{ EMPPUBLICA}$

(Law N<sup>o</sup> 6987 DE 11/05/2020). This law provides for the electronic processing of administrative processes in the Municipality of Maceió, supporting the protection of users.

Art 6. (about peticionamento): This law says that an electronic process is considered invalid if it has not a system date and time or has not the IP address of the station that originated the process [89].

$\vdash_{iALC} (w : (\neg \forall \text{possuiUma.DATAHORA} \sqcup \neg \forall \text{temIp.IPESTACAO}) \rightarrow (\neg \exists \text{tipoProcesso.VALIDO}))$

CQ<sub>13</sub> - Counterexample:

a)  $w'''' : \mathcal{F} \text{ DATAHORA} ; w''' : \mathcal{T} \text{ VALIDO}$

b)  $w'''' : \mathcal{F} \text{ IPESTACAO} ; w''' : \mathcal{T} \text{ VALIDO}$

(Lei N<sup>o</sup> 5671 DE 28/12/2007). Gratificação de Produtividade Ambiental (GPA).

Art. 1 - The Gratificação de Produtividade Ambiental (GPA) is hereby established, which will be attributed to employees assigned to the Secretaria Municipal de Proteção ao Meio Ambiente (SEMPMA), who contribute with effective and proven participation in support and control activities, technical support, monitoring and inspection of the environment.

§ 2 - The score referred to in this article will comply with the limit of 100 (one hundred) points for employees classified in Group I, 60 (sixty) points for employees classified in Group II and 30 (thirty) points for employees classified in Group III.

$\vdash_{iALC} (w : ((\forall \text{emLotacao.SEMPMA}) \sqcap (\forall \text{temGrupo.GI})) \rightarrow (\neg (\exists \text{temScore.SCORE100}) \sqcap \neg (\exists \text{temScore.SCORE60})))$

CQ<sub>14</sub> - Counterexample:

a)  $w''' : \mathcal{T} \text{ SCORE100} ; w' : \mathcal{T} \text{ GI} ; w' : \mathcal{T} \text{ SEMPMA}$

b)  $w''' : \mathcal{T} \text{ SCORE60} ; w' : \mathcal{T} \text{ GI} ; w' : \mathcal{T} \text{ SEMPMA}$

$\vdash_{iALC} (w : ((\forall \text{emLotacao.SEMPMA}) \sqcap (\neg \forall \text{temGrupo.GI})) \rightarrow ((\exists \text{temScore.SCORE60}) \sqcap (\exists \text{temScore.SCORE100})))$

CQ<sub>15</sub> - Counterexample:

a)  $w'' : \mathcal{F} \text{ GI} ; w' : \mathcal{F} \text{ SCORE60} ; w' : \mathcal{T} \text{ SEMPMA}$

b)  $w'' : \mathcal{F} \text{ GI} ; w' : \mathcal{F} \text{ SCORE100} ; w' : \mathcal{T} \text{ SEMPMA}$

(Lei N<sup>o</sup>. 7249 DE 02/06/2011). Gratificação de Produtividade Secretaria de Gestão (SEMGE).

Paragraph 1<sup>o</sup> - Scale D will be assigned a maximum value of 40 points obtained from the sum of the evaluation factors, with a maximum value of 35 points for Scale C, a maximum value of 30 points for Scale B, and a maximum value of 25 points for Scale A.

$$\vdash_{iALC} (w : ((\forall emLotacao.SEMGE) \sqcap (\forall emEscala.ESCALAA)) \multimap ((\exists temScore.SCORE25)))$$

CQ<sub>16</sub> - Counterexample:

$w' : \mathcal{T} ESCALAA ; w' : \mathcal{T} SEMGE ; w' : \mathcal{F} SCORE25$

(Lei N<sup>o</sup>. 6685 DE 18/08/2017). Law that establishes the Maceió Tax Code (SEFAZ) Art. 3<sup>o</sup> The tributes that are part of the Municipal Tax Code are: I - Taxes and II - Fees [90].

The Location License Fee (LLF) is the fee due for the municipal activity verifying compliance with legislation regulating the use and occupation of urban ground.

Some LLF exemption rules:

- Legally constituted sports association:

$$\vdash_{iALC} (w : ((\forall tipoEmpresa.ASSDESP) \sqcap (\neg(\forall temRegistro.LEGAL)) \multimap (\neg(\exists temBeneficio.ISENCAO))))$$

CQ<sub>17</sub> - Counterexample:

$w''' : \mathcal{F} LEGAL ; w''' : \mathcal{T} ISENCAO ; w' : \mathcal{T} ASSDESP$

Another approach to CQ<sub>17</sub>:

$$\vdash_{iALC} (w : ((\forall tipoEmpresa.ASSDESP) \sqcap ((\forall temRegistro.LEGAL)) \multimap ((\exists temBeneficio.ISENCAO))))$$

CQ<sub>18</sub> - Counterexample:

$w' : \mathcal{T} LEGAL ; w' : \mathcal{T} ASSDESP ; w' : \mathcal{F} ISENCAO$

- Being social assistance, philanthropic or charitable entities:

$$\vdash_{iALC} (w : ((\forall tipoEmpresa.ASSSOCIAL) \sqcap (\neg(\forall temFinalidade.FILANTROPO) \sqcap (\neg(\forall temFinalidade.BENEFICENTE)) \multimap (\neg(\exists temBeneficio.ISENCAO))))$$

CQ<sub>19</sub> - Counterexample:

$w''' : \mathcal{F}$  BENEFICENTE ;  $w'' : \mathcal{F}$  FILANTROPO ;  $w''' : \mathcal{T}$  ISENCAO ;  $w' : \mathcal{T}$  ASSSOCIAL

- The public entities of direct administration and their respective “autarquias”.

$\vdash_{iALC} (w : ((\forall \text{tipoEmpresa}.EMPPUBLICA) \sqcap$   
 $(\neg(\forall \text{estaOrganograma}.ADMDIRETA)) \multimap \neg(\exists \text{temBeneficio}.ISENCAO)))$

CQ<sub>20</sub> - Counterexample:

$w''' : \mathcal{F}$  ADMDIRETA ;  $w''' : \mathcal{T}$  ISENCAO ;  $w' : \mathcal{T}$  EMPPUBLICA

(Acórdão N<sup>o</sup> 3561/23). Public agents perform essential functions. Requirements and qualifications expressly provided for by law, in accordance with this decision. Prejudged N<sup>o</sup> 25-TCE/PR. It is prohibited for occupants of a commissioned position to receive a bonus function (Função Gratificada)[...]

It is not possible to accumulate remuneration for a commission position with a bonus function [...] N<sup>o</sup> 25 [91].

$\vdash_{iALC} w : ((\forall \text{temCargo}.CC) \multimap (\exists \text{temCargo}.CC \sqcap \neg(\exists \text{temCargo}.FG)))$

CQ<sub>21</sub> - Counterexample:

$w''' : \mathcal{T}$  FG ;  $w' : \mathcal{T}$  CC

## Appendix B

In this appendix, we present a grammar of the ALC language using the syntax of the ANTLR4 tool. In fact, this grammar extends to an attribute grammar. As we can see, in line 16, the attribute grammar mechanism adds a layer of functionality, allowing the association of semantic information (*left=formula*, *op=binconnective*, *right=formula*) with the original grammar elements.

```

1  /*
2  * iALC written for Antlr4 by Fernando;Hermann
3  */
4  grammar iALCGrammar;
5  @header {
6      package iALC.grammar;
7  }
8
9  s : condition EOF;
10 condition
11     : formula (ENDLINE formula)* ENDLINE* EOF
12     ;
13
14 formula
15     : LPAREN* atom=CHARACTER RPAREN* #atomFormula
16     | left=formula op=bin_connective right=formula #opFormula
17     | LPAREN formula RPAREN #parenFormula
18     | NOT formula #notFormula
19     | FORALL ROLE '.' (LPAREN* CHARACTER bin_connective*
20     CHARACTER* RPAREN*) #forallFormula
21     | EXISTS ROLE '.' (LPAREN* CHARACTER bin_connective*
22     CHARACTER* RPAREN*) #existFormula
23     | modal #forallAtomico
24     ;
25
26 asser
27     : CHARACTER* '.' formula
28     ;
29
30 ROLE
31     : CHARACTER*
32     ;
33
34 LPAREN

```

```
33      : '('
34      ;
35 RPAREN
36      : ')'
37      ;
38
39 modal
40      : FORALL
41      | EXISTS
42      ;
43
44
45 FORALL
46      : '\u2200'
47      ;
48 EXISTS
49      : '\u018E'
50      ;
51 bin_connective
52      : CONJ
53      | DISJ
54      | IMPL
55      ;
56
57 NOT
58      : '~'
59      ;
60
61 CONJ
62      : '\u2293'
63      ;
64
65 DISJ
66      : '\u2294'
67      ;
68 IMPL
69      : '\u226B'
70      ;
71
72 CHARACTER
73      : [0-9a-zA-Z]+
```

```
74      ;
75
76 ENDLINE
77      : ( '\r' | '\n' ) +
78      ;
79 WHITESPACE
80      : ( ' ' | '\t' ) + -> skip
81      ;
```

Code Listing 8.1: Exemplo de código ANTLR4



## Appendix C

---

### Listing 23 R program for calculating accuracy;

---

```
# Vetor de respostas corretas
respostas_das_CQs <- list(c("T-IRPF", "T-INCAPAC", "T-APOSENT"),
  c("T-RPPS", "T-COMISSONADO", "T-SERVIDOR"),
  c("F-EST", "T-SERVIDOR", "F-RGPS"),
  c("F-EST", "F-CEL", "T-SERVIDOR", "F-EST", "F-RGPS", "T-SERVIDOR"),
  c("F-CEL", "F-EST", "T-SERVIDOR", "F-CEL", "F-RPPS", "T-SERVIDOR"),
  c("T-TECNICOB", "T-ACUMULAR", "T-TECNICOA"),
  c("T-ACUMULAR", "T-TECNICOB", "T-TECNICOA"),
  c("T-TECNICOA", "T-MAGISTERIO", "F-ACUMULAR"),
  c("T-EFETIVO", "T-MENOR2ANOS", "T-CONCURSADO"),
  c("F-PESQUISADOR", "F-DEZANOS", "T-ATIVO"),
  c("T-ATIVO", "T-MENOS3ANOS", "T-EMPPUBLICA"),
  c("T-MENOS3ANOS", "T-ATIVO", "T-EMPPUBLICA"),
  c("F-DATAHORA", "T-VALIDO", "F-IPESTACAO", "T-VALIDO"),
  c("T-SCORE100", "T-GI", "T-SEMPMA", "T-SCORE60", "T-GI", "T-SEMPMA"),
  c("F-GI", "F-SCORE60", "T-SEMPMA", "F-GI", "F-SCORE100", "T-SEMPMA"),
  c("T-ESCALAA", "T-SEMGE", "F-SCORE25"),
  c("F-LEGAL", "T-ISENCAO", "T-ASSDESP"),
  c("T-LEGAL", "T-ASSDESP", "F-ISENCAO"),
  c("F-BENEFICIENTE", "F-FILANTROPO", "T-ISENCAO", "T-ASSSOCIAL"),
  c("F-ADMDIRETA", "T-ISENCAO", "T-EMPPUBLICA"),
  c("T-FG", "T-CC"))

# Vetor de respostas do KRM
respostas_do_SAT <- list(c("T-IRPF", "T-INCAPAC", "T-APOSENT"),
  c("T-RPPS", "T-COMISSONADO", "T-SERVIDOR"),
  c("F-EST", "T-SERVIDOR", "F-RGPS"),
  c("F-EST", "F-CEL", "T-SERVIDOR", "F-EST", "F-RGPS", "T-SERVIDOR"),
  c("F-CEL", "F-EST", "T-SERVIDOR", "F-CEL", "F-RPPS", "T-SERVIDOR"),
  c("T-TECNICOB", "T-ACUMULAR", "T-TECNICOA"),
  c("T-ACUMULAR", "T-TECNICOB", "T-TECNICOA"),
  c("T-TECNICOA", "T-MAGISTERIO", "F-ACUMULAR"),
  c("T-EFETIVO", "T-MENOR2ANOS", "T-CONCURSADO"),
  c("F-PESQUISADOR", "F-DEZANOS", "T-ATIVO"),
  c("T-ATIVO", "T-MENOS3ANOS", "T-EMPPUBLICA"),
  c("T-MENOS3ANOS", "T-ATIVO", "T-EMPPUBLICA"),
  c("F-DATAHORA", "T-VALIDO", "F-IPESTACAO", "T-VALIDO"),
  c("T-SCORE100", "T-GI", "T-SEMPMA", "T-SCORE60", "T-GI", "T-SEMPMA"),
  c("F-GI", "F-SCORE60", "T-SEMPMA", "F-GI", "F-SCORE100", "T-SEMPMA"),
  c("T-ESCALAA", "T-SEMGE", "F-SCORE25"),
  c("F-LEGAL", "T-ISENCAO", "T-ASSDESP"),
  c("T-LEGAL", "T-ASSDESP", "F-ISENCAO"),
  c("F-BENEFICIENTE", "F-FILANTROPO", "T-ISENCAO", "T-ASSSOCIAL"),
  c("F-ADMDIRETA", "T-ISENCAO", "T-EMPPUBLICA"),
  c("T-FG", "T-CC"))

# Verificar se as respostas estão corretas
respostas_corretas <- unlist(respostas_das_CQs) == unlist(respostas_do_SAT)

# Calcular a acurácia
acuracia <- sum(respostas_corretas) / length(respostas_corretas)

# Exibir o resultado
print(paste("Acurácia:", acuracia))
```

---

## Appendix D

This appendix shows the soundness proof of the algorithm that implements the calculi system of the *iALC* tableau, Algorithm (1).

**Definition 8.1** (*A Complete Tableau*). Let  $\theta$  be a branch and  $\alpha$  and  $\beta$  be *iALC* formulas. A tableau  $\Upsilon$  is Complete if for every  $\alpha \in \theta$ , both  $\alpha_1, \alpha_2 \in \theta$ . Another way, for every  $\beta \in \theta$ , either  $\beta_1 \in \theta$  or  $\beta_2 \in \theta$ .

**Theorem 8.1** (Soundness of *iALC* Algorithm). Let  $\delta$  be a valid formula of *iALC*,  $\Gamma$  a set of *iALC* formulas, and  $\Gamma \in \Upsilon$ . Then,  $\Gamma \vdash_{tab} \delta$ .

**Lemma 8.1** (*Alpha-Type<sub>[F(→)]</sub> Formulas*). Let  $\theta$  be a branch and  $\alpha$  and  $\beta$  be *iALC* formulas,  $w \in W$  and  $(\alpha, \beta) \in \theta$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $S, \mathcal{T}(w : (\alpha \rightarrow \beta))$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $S, \mathcal{T}(w : (\alpha \rightarrow \beta))$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $S, \mathcal{T}(w : (\alpha \rightarrow \beta))$  for  $\Gamma \in \theta$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \models w : (\alpha \rightarrow \beta)$ . Then  $\Gamma \models (w' : \alpha)$  and  $\Gamma \not\models (w' : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{T}(w' : \alpha)\}$  and  $\{S, \mathcal{F}(w' : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w \mathcal{R} w'$ , Definition (5.3), for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.2** (*Beta-Type<sub>[T(→)]</sub> Formulas*). Let  $\theta$  be a branch and  $\alpha$  and  $\beta$  be *iALC* formulas,  $w \in W$  and  $\alpha \in \theta$  or  $\beta \in \theta$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $S, \mathcal{T}(w : (\alpha \rightarrow \beta))$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $S, \mathcal{T}(w : (\alpha \rightarrow \beta))$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $S, \mathcal{T}(w : (\alpha \rightarrow \beta))$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \models w : (\alpha \rightarrow \beta)$ . Then  $\Gamma \not\models (w_L : \alpha)$  with  $\alpha \in \theta$  and  $\Gamma \models (w_R : \beta)$  with  $\beta \in \theta'$ , so either  $\Gamma$  realizes  $\{S, \mathcal{F}(w : \alpha)\}$  and  $\{S, \mathcal{T}(w : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w_L \preceq w_R$ , for some  $w_L \in \theta$  and  $w_R \in \theta'$  ( $w_s \in W$ ).

**Lemma 8.3** (*Beta-Type<sub>[T(⊔)]</sub> Formulas*). Let  $\theta$  be a branch,  $\alpha$  and  $\beta$  be *iALC* formulas,  $w \in W$ ,  $\alpha \in \theta$  or  $\beta \in \theta$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w : \alpha)\}, \{S, \mathcal{T}(w : \beta)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $\{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{T}(w : (\alpha \sqcup \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \models w : (\alpha \sqcup \beta)$ . Then  $\Gamma \models (w_L : \alpha)$  or  $\Gamma \models (w_R : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{T}(w_L : \alpha)\}$  or  $\{S, \mathcal{T}(w_R : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w_L \preceq w_R$ , for some  $w_L \in \theta$  and  $w_R \in \theta'$  ( $w_s \in W$ ).

**Lemma 8.4** (*Alpha-Type<sub>[ $\mathcal{F}(\sqcup)$ ]</sub>* Formulas). *Let  $\theta$  be a branch,  $\alpha$  and  $\beta$  be iALC formulas,  $w \in W$  and  $(\alpha, \beta) \in \theta$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w : \alpha), \mathcal{F}(w : \beta)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{i+1}$ .*

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $\{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{F}(w : (\alpha \sqcup \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \not\models w : (\alpha \sqcup \beta)$ . Then  $\Gamma \not\models (w' : \alpha)$  and  $\Gamma \not\models (w' : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{F}(w' : \alpha)\}$  or  $\{S, \mathcal{F}(w' : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w \preceq w'$ , for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.5** (*Alpha-Type<sub>[ $\mathcal{T}(\sqcap)$ ]</sub>* Formulas). *Let  $\theta$  be a branch,  $\alpha$  and  $\beta$  be iALC formulas,  $w \in W$  and  $(\alpha, \beta) \in \theta$ . For  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w : \alpha), \mathcal{T}(w : \beta)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{i+1}$ .*

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $\{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{T}(w : (\alpha \sqcap \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \models w : (\alpha \sqcap \beta)$ . Then  $\Gamma \models (w : \alpha)$  and  $\Gamma \models (w : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{T}(w' : \alpha)\}$  and  $\{S, \mathcal{T}(w' : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w \preceq w'$ , for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.6** (*Beta-Type<sub>[ $\mathcal{F}(\sqcap)$ ]</sub>* Formulas). *Let  $\theta$  be a branch,  $\alpha$  and  $\beta$  be iALC formulas,  $w \in W$ ,  $\alpha \in \theta$  or  $\beta \in \theta$ , and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w : \alpha)\}, \{S, \mathcal{F}(w : \beta)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{i+1}$ .*

**Proof** Since  $\delta_i$  is realizable, some element of it is realizable. If that element is not  $\{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}$ , the same element of  $\delta_{i+1}$  is realizable. If that element is  $\{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}$ , then for some model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma$  realizes  $\{S, \mathcal{F}(w : (\alpha \sqcap \beta))\}$ . Consequently,  $\Gamma$  realizes  $S$  and  $\Gamma \not\models w : (\alpha \sqcap \beta)$ . Then  $\Gamma \not\models (w_L : \alpha)$  and  $\Gamma \not\models (w_R : \beta)$ , so either  $\Gamma$  realizes  $\{S, \mathcal{F}(w_L : \alpha)\}$  and  $\{S, \mathcal{F}(w_R : \beta)\}$ . In either case,  $\delta_{i+1}$  is realizable. Where  $w_L \preceq w_R$ , for some  $w_L \in \theta$  and  $w_R \in \theta'$  ( $w_s \in W$ ).

**Lemma 8.7** (*Alpha-Type<sub>[ $\mathcal{F}(\neg)$ ]</sub>* Formulas). *Let  $\theta$  be a branch,  $\alpha$  an iALC formula,  $w \in W$ ,  $\alpha \in \theta$  and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : \neg\alpha)\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w' : \alpha)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{i+1}$ .*

**Proof** Since  $\delta_i$  is realizable, and it suffices to consider the case that  $\{S, \mathcal{F}(w : \neg\alpha)\}$  is the realizable element. Then there is a model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and a  $\Gamma \in \Delta^{\mathcal{I}}$  such that  $\Gamma$  realizes  $S$  and  $\Gamma \not\models (w : \neg\alpha)$ . Since  $\Gamma \not\models (w : \neg\alpha)$ , for some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma \models \delta$ . But clearly, if  $\Gamma$  realizes  $S$ ,  $\Gamma^*$  (all  $\Gamma$ ) realizes  $w'$  (for a new  $w$ ) and  $[w \prec w']$  by definition (5.4). Hence  $\Gamma^*$  realizes  $\{S, [\mathcal{T}(w \prec w')], \mathcal{T}(w' : \alpha)\}$  and  $\delta_{i+1}$  is realizable. Where  $w \mathcal{R} w'$ , Definition (5.3), for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.8** (*Alpha-Type<sub>[ $\mathcal{T}(\neg)$ ]</sub>* Formulas). *Let  $\theta$  be a branch,  $\alpha$  an iALC formula,  $w \in W$ ,  $\alpha \in \theta$  and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : \neg\alpha)\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w' :$*

$\alpha\}\dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** Since  $\delta_i$  is realizable, and it suffices to consider the case that  $\{S, \mathcal{T}(w : \neg\alpha)\}$  is the realizable element. Then there is a model  $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \preceq \rangle$  and a  $\Gamma \in \Delta^{\mathcal{I}}$  such that  $\Gamma$  realizes  $S$  and  $\Gamma \models (w : \neg\alpha)$ . Since  $\Gamma \models (w : \neg\alpha)$ , for some  $\Gamma \in \Delta^{\mathcal{I}}$ ,  $\Gamma \models \delta$ . But clearly, if  $\Gamma$  realizes  $S$ ,  $\Gamma^*$  (all  $\Gamma$ ) realizes  $w'$  and  $[w \prec w']$  by definition (5.4). Hence  $\Gamma^*$  realizes  $\{S, [\mathcal{F}(w \prec w'), \mathcal{F}(w' : \alpha)]\}$  and  $\delta_{i+1}$  is realizable. Where  $w \preceq w'$ , for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.9** (Alpha-Type $_{[T(\exists R.C)]}$  Formulas). Let  $\theta$  be a branch,  $\alpha$  an iALC formula,  $w \in W$ ,  $\alpha \in \theta$  and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : (\exists R.C))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w' : C)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** if  $S, \mathcal{T}(w : (\exists R.C))$  is realizable, and if  $w', w' \in W$  does not occur in  $S$ , then  $S, \mathcal{T}(w' : C)$  is realizable. Where  $w \mathcal{R} w'$ , Definition (5.3), for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.10** (Alpha-Type $_{[F(\exists R.C)]}$  Formulas). Let  $\theta$  be a branch,  $\alpha$  an iALC formula,  $w \in W$ ,  $\alpha \in \theta$  and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : (\exists R.C))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w' : C)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** if  $S, \mathcal{F}(w : (\exists R.C))$  is realizable, and if  $w', w' \in W$  occur in  $S$ , then  $S, \mathcal{F}(w' : C)$  is realizable. Where  $w \preceq w'$ , for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.11** (Alpha-Type $_{[T(\forall R.C)]}$  Formulas). Let  $\theta$  be a branch,  $\alpha$  an iALC formula,  $w \in W$ ,  $\alpha \in \theta$  and  $\delta_i$  is  $\{\dots, \{S, \mathcal{T}(w : (\forall R.C))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{T}(w' : C)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** if  $S, \mathcal{T}(w : (\forall R.C))$  is realizable, and if  $w', w' \in W$  occur in  $S$ , then  $S, \mathcal{T}(w' : C)$  is realizable. Where  $w \preceq w'$ , for some  $(w, w') \in \theta$  ( $w^i \in W$ ).

**Lemma 8.12** (Alpha-Type $_{[F(\forall R.C)]}$  Formulas). Let  $\theta$  be a branch,  $\alpha$  an iALC formula,  $w \in W$ ,  $\alpha \in \theta$  and  $\delta_i$  is  $\{\dots, \{S, \mathcal{F}(w : (\forall R.C))\}, \dots\}$  and  $\delta_{i+1}$  is  $\{\dots, \{S, \mathcal{F}(w' : C)\}, \dots\}$ . If  $\delta_1$  is realizable (Theorem 5.1), so is  $\delta_{1+1}$ .

**Proof** if  $S, \mathcal{F}(w : (\forall R.C))$  is realizable, and if  $w', w' \in W$  does not occur in  $S$ , then  $S, \mathcal{F}(w' : C)$  is realizable. Where  $w \mathcal{R} w'$ , Definition (5.3), for some  $(w, w') \in \theta$  ( $w^i \in W$ ).