Using Grammar-Based Genetic Programming for Mining Subsumption Axioms Involving Complex Class Expressions

Rémi Felin, Andrea G. B. Tettamanzi

Using Grammar-Based Genetic Programming for Mining Subsumption Axioms Involving Complex Class Expressions

Remi Felin
Andrea G. B. Tettamanzi
remi.felin@etu.univ-cotedazur.fr
andrea.tettamanzi@univ-cotedazur.fr
Université Côte d'Azur, Inria, CNRS, I3S
Sophia Antipolis, France

ABSTRACT

Ontology enrichment is a key task in the area of the Semantic Web. It allows directly to enrich the links between entities of the semantic Web and thus adding information. Our research area is part of this objective with the search for axioms using an evolutionary process. We propose an adaptation resulting from the coupling between evolutionary algorithms and the theory of possibilities in order to allow the research of axioms of subsumption composed of complex classes.

CCS CONCEPTS

• **Computing methodologies** → **Ontology engineering**; • **Evolutionary algorithms**; • **Possibility Theory**;

KEYWORDS

Ontology Learning, OWL 2 Axiom, Subsumption Axioms, Grammatical Evolution

ACM Reference Format:
Remi Felin and Andrea G. B. Tettamanzi. 2021. Using Grammar-Based Genetic Programming for Mining Subsumption Axioms Involving Complex Class Expressions. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI-IAT '21), December 14–17, 2021, ESSENDON, VIC, Australia.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3486622.3494025

1 INTRODUCTION

Over time, the semantic Web has seen major improvements, notably around the development of **Linked Open Data** (LOD) which can be seen as a database of related entities. The data is expressed through the **Resource Description Framework** (RDF) format, which is a Word Wide Web Consortium (W3C) standard. The ontologies are used to describe the representation of our knowledge graph and the **Web Ontology Language** (OWL) format is a language for building these representations and has become a W3C standard. In our work, we work with the OWL 2 format which is also standardised.

Ontology Learning has an important place in the development of the LOD as it allows the enrichment of ontologies through a set of research domains (see Section 2). Continuous learning is particularly important when using dynamic data sources where a large amount of data is subject to real-time changes. Therefore, continuous learning and inference of new knowledge, through consistency evaluation, seems to be a major issue in ontology enrichment. Despite the massive development of knowledge and LOD with the purpose of information extraction and formatting systems, there is a certain lack of information in ontologies and the quality of information present in the semantic Web is not guaranteed. Indeed, if we take one of the reference bases of the LOD, namely DBPedia, which is a knowledge base constructed based on information extracted from Wikipedia, it presents some shortcomings in terms of information, especially in the axiom sets of the ontology. Our research area focuses on **Axiom Learning** [5], which is a bottom-up approach, using learning algorithms and relying on instances from several existing knowledge and information resources to discover axioms. Axiom learning algorithms can help reduce the overall cost of axiom extraction and ontology construction in general. To this aim, we use an evolutionary approach, namely **Grammatical Evolution** [3]. Using a predefined grammar in BNF format (see Section 2), we can generate candidate axioms, formed according to the syntax defined in the **BNF** file, at random. Of course, this simple process is not sufficient to obtain an axiom that is meaningful. For this purpose, we use an evolutionary process based on this grammar to allow the generation of random candidate axioms, which together form a population, and the evaluation of this population using a *fitness* function, which we aim to maximise. This process, which iterates according to the given parameters, allows us to obtain from initially random candidate axioms, new candidate axioms that are more and more consistent, i.e., which present a non-zero fitness, and these individuals will be taken as model by the algorithm to generate a new population of axioms which inherit traits from the best individuals. In order to be able to evaluate candidate axioms satisfactorily, we base our heuristic on possibility theory [6] to calculate the **possibility** and **necessity** for a given axiom. Indeed, this theory allows us to overcome the lack of information in the datasets and the unguaranteed quality which, with a probabilistic heuristic, has a negative impact on the results.

In this paper, we will focus on the extraction of complex class subsumption axioms. In particular, we will be interested in complex classes constructed with relational operators of existential quantification or value restriction, like $\exists r.C$ and $\forall r.C$ where C is an atomic class and r is a property. Before discussing the tools used and the experimental protocol, we will briefly review the different concepts of the semantic Web, and discuss the evolutionary algorithm used for mining these axioms. Finally, we present the results followed by an analysis and discussion.

WI-IAT '21, December 14–17, 2021, ESSENDON, VIC, Australia

2 PRELIMINARIES

2.1 Web of Data

While the semantic Web allows the sharing and distribution of data in the form of knowledge across the Web, one of its objectives is also to allow intelligent agents to infer new knowledge from the knowledge already acquired from the Web. The RDF standard defined earlier allows for the linking of data to each other, especially in distribution but also in form, while allowing for accessibility. RDF statements are structured as *triples* of the form ⟨*Subject Predicate Object*⟩, where each part of the triple consists of an IRI, and the object can also consist of a *literal*. The framework is based on the XML format, a tagging language that allows, among other things, a great flexibility concerning the encoding of data. Each abstract or physical resource is accessible through its respective **International Resource Identifier** (IRI) in order to comply with data accessibility constraints. The query language for manipulating RDF data across the net (through a given endpoint) is the **SPARQL** language, which is a W3C recommendation.

As discussed in [2], the term **ontology** has several definitions from both a philosophical and a computational perspective.

> *An ontology is a formal and explicit specification of a shared conceptualization*

is one of the most relevant definitions discussed, where the conceptualization are an abstract representation of the knowledge of universe under consideration, also called the **universe of discourse**, in which concepts and entities are represented, as well as the meaning of their relations in this same universe.

In the context of our research, we consider an **Ontology** (O) as a quadruple

$$O = \langle C, \mathcal{R}, \mathcal{I}, \mathcal{A} \rangle,$$

where C is a set of concepts, $\mathcal{R}$ is a set of relations, $\mathcal{I}$ is a set of instances, i.e. instances, in which two or more concepts are related to each other, $\mathcal{A}$ is a set of axioms.

In order to respond to the enrichment of knowledge, we need to be able to translate information already present in terms of meaning and to be able to evaluate the quality of this information before deducing new information. These sets of objectives are part of **Ontology Learning**.

This area includes different distinct sub-areas [7] for continuous enrichment of ontology processes such as learning ontologies from text, which is part of the *Natural Language Processing* area. Learning schema axioms, such as definitions of classes, from ontologies and instances data is a research area of the Concept Learning, or the use of crowdsourcing which is a purely automatic approach. The area on which we focus our efforts is the extraction of information from RDF graphs and therefore from already existing data but which may present both quantity and quality shortcomings.

2.2 OWL 2 Axioms

OWL 2 provides for 32 types of axioms [1], divided in 6 categories:

- **Class expression** axioms (SubClassOf, DisjointClasses, …)
- **Object property expression** axioms (SubObjectPropertyOf, DisjointObjectProperties, …)
- **Data property expression** axioms (SubDataPropertyOf, DisjointDataProperties, …)
- **Datatype definition** axioms (DatatypeDefinition)
- **Keys** axioms (HasKey)
- **Assertion** axioms (ClassAssertion, ObjectPropertyAssertion)

We have chosen to work with OWL 2 axioms in order to satisfy two objectives:

- Extracting new knowledge from an existing knowledge base expressed in RDF.
- Injecting such extracted knowledge into an ontology in order to be able to use it to infer its logical consequences.

The format is appropriate for these two distinct purposes as it is very expressive and is suitable for injection into an ontology.

2.3 Possibility Theory

In order to be able to evaluate the axioms taking into account the lack of information, and more precisely the difficulty to find counterexamples, essential to our calculation. We apply the Possibility theory. [6, 10]

Possibility theory is a mathematical theory of epistemic uncertainty which uses the events, variables, … denoted ω of a universe of discourse Ω ($\omega \in \Omega$) where each ω has a degree of possibility such that $\pi : \Omega \to [0, 1]$ where $\pi(\omega) = 0$ means that ω is not possible (excluded) and $\pi(\omega) = 1$ that ω is completely possible. The theory includes a measure of possibility denoted by Π and a measure of necessity denoted by N such that:

$$\begin{aligned} \Pi(A) &= \max_{\omega \in A} \pi(\omega), \\ N(A) &= 1 - \Pi(\bar{A}) = \min_{\omega \in \bar{A}} \{1 - \pi(\omega)\}, \end{aligned}$$

where $A \in \Omega$ or $A = \{\omega : \omega \models \phi\}$.

Some properties for $A \subseteq \Omega$:

- $\Pi(\emptyset) = N(\emptyset) = 0$
- $\Pi(\Omega) = N(\omega) = 1$
- $\Pi(A) = 1 - N(\bar{A})$ (duality)
- $\Pi(A) = 1$ if $N(A) > 0$
- $N(A) = 1$ if $\Pi(A) < 1$
- $\Pi(A) = \Pi(\bar{A}) = 1$ (complete ignorance on A)

3 FOUNDATIONS

In this section, we will present the work that has been done in the context of our research on axiom discovery in knowledge graphs.

3.1 Possibilistic OWL Axioms Scoring

This section is a review of [10], which has served as a basis for the work we will detail in the following section. The subsumption axioms, belonging to the category of class expression axioms, have the form *SubClassOf(CD)*; in their most basic form, involving atomic classes, C and D are simple classes represented by their respective IRI. This means that $C \subseteq D$ and therefore all resources in C and D verify the property $C^I \subseteq D^I$.

In order to be able to apply our calculations, the definition of the sets included in the scope of a axiom ϕ, which we will note *content*, will serve us as a necessary restricted context in order to evaluate ϕ, thus:

$$content(\phi) = \{\psi : \omega \models \psi\},$$

where every formula $\psi \in \text{content}(\phi)$ may be tested with a SPARQL *ASK* query. We can set v_ϕ as the support of axiom where:

$$v_\phi = |content(\phi)|.$$

In order to assess the possibility and necessity of an axiom, let us consider v_ϕ^+ the confirmations observed among the elements of v_ϕ and v_ϕ^- the conterexamples observed. We define the possibility $\Pi(\phi)$ and necessity $N(\phi)$ of an axiom as follows:

$$\Pi(\phi) = 1 - \sqrt{1 - \left(\frac{v_\phi - v_\phi^-}{v_\phi}\right)^2},$$

$$N(\phi) = \sqrt{1 - \left(\frac{v_\phi - v_\phi^+}{v_\phi}\right)^2},$$

if $\Pi(\phi) = 1$, 0 otherwise.

Once these indicators are obtained, we define an **Acceptance/Rejection Index** (ARI) as:

$$ARI(\phi) = N(\phi) - N(\neg\phi) = N(\phi) + \Pi(\phi) - 1 \in [-1, 1].$$

For this purpose, if the ARI is less than 0, it means that we reject the axiom, alternatively we can validate the axiom. When the ARI is equal to (or close to) 0, we are in an ignorance scenario for the axiom being tested.

Thus, the SPARQL queries constructed in order to retrieve confirmations (1) and counterexamples (2) are as follows:

```
(1) SELECT (count(DISTINCT ?x) AS ?numConfirmations)
    WHERE { Q(C, ?x) Q(D, ?x) }

(2) SELECT (count(DISTINCT ?x) AS ?numExceptions)
    WHERE { Q(C, ?x) Q(¬D, ?x) }
```

Where Q is a mapping from OWL 2 class expressions to SPARQL graph patterns, the first part of the component is an OWL 2 expression and the last one a variable [8]. For instance, for (1) we can write it as:

```
SELECT (count(DISTINCT ?x) as ?numConfirmations) WHERE {
    ?x a C .
    ?x a D
}
```

and (2) as:

```
SELECT (count(DISTINCT ?x) as ?numExceptions) WHERE {
    ?x a C .
    ?x a ?y .
    FILTER NOT EXISTS {
        ?z a ?y .
        ?z a D
    }
}
```

3.2 Grammatical Evolution

Before talking about the general functioning of Grammatical Evolution, it is important to specify the prerequisites to do so. Indeed, the algorithm bases its axiom generation on a **training dataset** (or a minimized dataset) extracted from the original dataset from which we wish to discover axioms, this aims at reducing the field of possibilities as for the axiom generation but also at saving time on the evaluation of the candidate axioms, which represents a very expensive operation in terms of computation time.

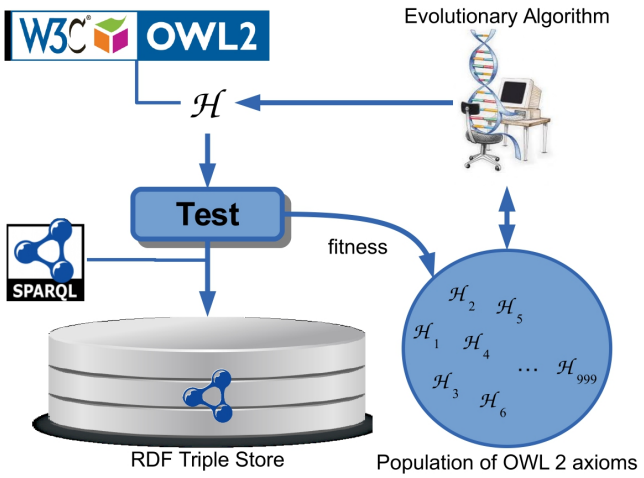

Figure 1: A schematic illustration of the proposed approach.

Grammatical Evolution is based on a classical evolutionary process comprising the cycles of generation of a population, fitness assessment for each individual in the population, selection of the best individuals from the population, mutation and crossover (the triggering of which depends on a random variable) of the least well assessed individuals. The generation of a population is based on a grammar defined beforehand with the help of a BNF file allowing to represent the structure of the axioms that one wishes to generate while respecting the constraints of *generality* and *credibility*. The BNF grammar is divided into two distinct parts [4]: a static part, which contains the rules concerning the modelling of our axioms (see Section 4) and a dynamic part which is filled in during the execution of our software. An example, taken from [1], of a rule commonly found in this file is the following:

```
(r.1) ClassAxiom := SubClassOf | EquivalentClasses |
DisjointClasses | DisjointUnion
(r.2) SubClassOf := 'SubClassOf' '(' subClassExpression '
' superClassExpression ')'
(r.3) subClassExpression := ClassExpression
(r.4) superClassExpression := ClassExpression
(r.5) ClassExpression := Class
```

The dynamic part contains the rules extracted from our training dataset and thus containing resources useful to the previous rules established, so we use SPARQL queries to retrieve productions from the training dataset, which are also called primitives. In our work, we use production rules to obtain **Class** (3) and **ObjectPropertyOf** (4) whose queries are the following:

```
(3) SELECT ?class WHERE { ?instance rdf:type ?class.}

(4) SELECT ?property WHERE { ?subject ?property ?object.
    FILTER(isIRI(?object)) }
```

Thus, we retrieve the resources returned by the queries and write to our BNF file, the resources returned as:

```
(r.6) Class := dbo:Agent
            |  dbo:Book
            |  dbo:Document
            ...
(r.7) ObjectPropertyOf := dbp:leaderName
                       |  dbp:extendedFrom
                       |  dbp:birthDate
                       ...
```

The individuals of a population are formed from the previously established rules, thus giving a syntactically correct axiom in OWL format, but this axiom is not exploitable as such by the algorithm, which only manipulates numerical entities. These individuals are represented through a numerical integer sequence that represents a single axiom according to the production rules. We invite the reader to refer to [5] concerning the basic concept of grammar-based genetic programming.

3.3 Axiom Evaluation

In order to assess the fitness of our axioms and thus give them their value as individuals in the population. We apply the formulas previously studied with the possibility theory such as:

$$f(\phi) = v_\phi \times \frac{\Pi(\phi) + N(\phi)}{2}.$$

This formula allows us to value axioms for which there is a non-zero possibility and a zero necessity. Therefore, the new populations will be based on individuals with non-zero fitness, which avoids, as the algorithm is generated, the generation of axioms for which v_ϕ is zero and the possibility is greater than 0.

4 CONTRIBUTIONS

4.1 BNF Grammar Construction

In order to proceed with the experiments concerning complex class axiom extraction, we need to establish the static rules translated in the BNF file in order to obtain a subsumption axiom containing at least one complex class. This means an axiom of the form:

```
SubClassOf ( A B )
```

where either of A and B can be of the form:

```
ObjectSomeValuesFrom ( ObjectPropertyOf Class )
ObjectAllValuesFrom ( ObjectPropertyOf Class )
ObjectIntersection ( Class Class )
Class
```

where `Class` stands for an atomic class. For this purpose, the static rules of the grammar are crafted as follows:

```
(r.1) Axiom := ClassAxiom
(r.2) ClassAxiom := SubClassOf
(r.3) SubClassOf := 'SubClassOf' '(' classExpression '
      ' classExpression ')'
(r.4) classExpression := ObjectSomeValuesFrom |
    ObjectAllValuesFrom | ObjectIntersectionOf | Class
(r.5) ObjectIntersectionOf := 'ObjectIntersectionOf' '('
      Class ' ' Class ')'
(r.6) ObjectSomeValuesFrom := 'ObjectSomeValuesFrom' '('
      ObjectPropertyOf ' ' Class ')'
(r.7) ObjectAllValuesFrom := 'ObjectAllValuesFrom' '('
      ObjectPropertyOf ' ' Class ')'
```

The dynamic rules recovered for the experiments are those obtained with queries (3) and (4).

4.2 Merging and packaging of software

Within the framework of the directed work in the realization of a software allowing the mining of axiom [10] and the evaluation of axioms using the theory of the possibilities, this version was developed in order to evaluate axioms of the type *SubClassOf*. The second version of this software was the object of the work available in this source: [12], this one takes again the bases of this project by adding the grammatical evolution and the mining of disjointness axioms involving atomic classes [3] and complex classes [4].

The aim was to share these two pieces of software in order to be able to apply the grammatical evolution on any type of axiom and in particular the axioms of subsumption. The technical context presents a strong dependence on the Linux operating system with a set of libraries that can only be used on this system or on another system with the help of manipulation and transverse solutions that are costly in terms of resources and time. Furthermore, the installation process required to run the software is quite costly in terms of time and resources as it requires the installation of a local database to contain the training dataset useful for the experiments. We use **Virtuoso 7.2.5** in order to be able to store and manipulate our training dataset.

We propose to embed all our tools as well as the Virtuoso database management system in a global architecture using Docker technology. The sharing of the software in a capsule executable on any type of server will greatly help the development and use of the latter. Moreover, the installation and deployment of these tools are greatly simplified, saving us time and resources.

The source code of the project is available here [9].

5 EXPERIMENTS AND RESULTS

5.1 Experimental Protocol

In order to extract valid axioms respecting the format of the previously proposed grammar, we launched the software using grammatical evolution with the parameters indicated in Table 1.

Parameters such as population size, total effort k, ... are inspired by the parameters chosen in the experiments conducted in this [3–5] whose results obtained with these parameters were very adequate; therefore, we assume that these are the parameters that allow us to obtain optimal results for our experiments. The crossover technique used in the experiments is *sub-tree crossover* : a technique where two individuals are represented as linked graphs, a sub-graph of one individual will be merged with the sub-graph of the other with a given probability rate. Concerning mutation, the standard method is used: *single-point mutation* where the selected part of the

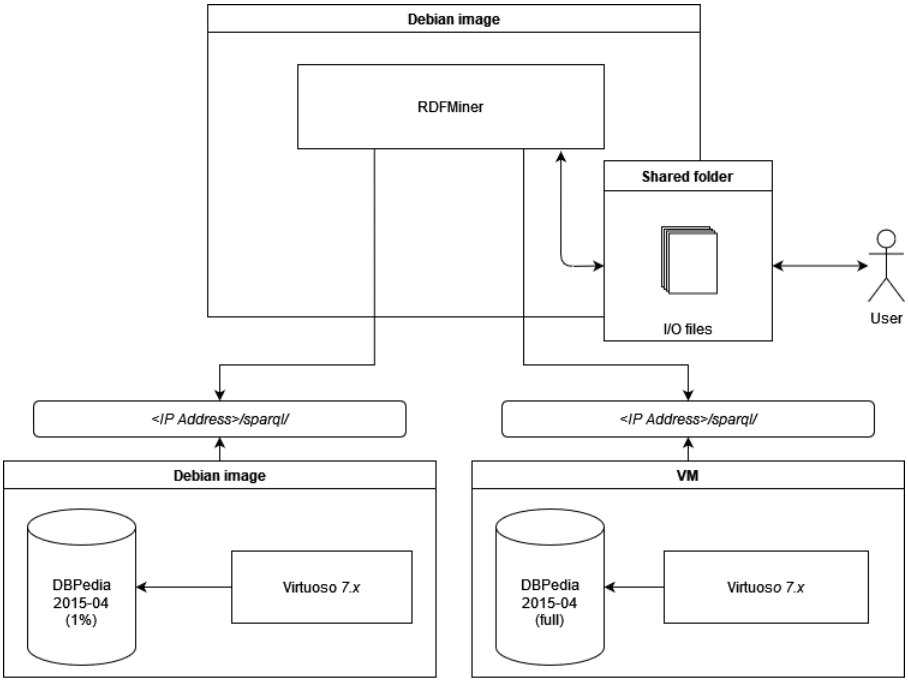

Figure 2: project architecture and interactions

Table 1: Parameters of Grammatical Evolution

Parameter	Values
popSize	100 ; 200 ; 500
Total effort: k	3000 ; 6000 ; 15000
initLenChrom	6
pCross	80%
pMut	1%
time-cap	30 seconds

individual is replaced by a new randomly generated with a given mutation rate [12].

Additionally, we use here a time limit (called *time-cap*) for the computation of counterexamples of Subsumption axioms, which represents a real problem in the evaluation. Indeed, the computation time using SPARQL queries, with our "open world" based heuristic, is very long and is directly related to the number of counterexamples that the query finds for each individual. One of the strategies is to use a time limit in order to reject axioms for which the computation of counterexamples would exceed the set time. We invite the reader to read the works discussing this method [11]. Given the number of individuals chosen in our experiments, we set a time limit of **30 seconds** which seems to represent a good compromise between a fair evaluation of the axioms and obtaining results in an acceptable time.

The evaluation function for our axioms is given in Section 3.3, which favors individuals for which we observe a non-zero number of classes concerned by the axiom and a non-zero possibility or necessity.

Concerning the exploited data, we use an image of the **DBPedia 2015-04** database on a **Virtuoso 7.2.5** server accessible via a SPARQL endpoint. We use the training dataset to perform the axiom extraction and the generations are from this same dataset, it represents **1%** of the entire DBPedia 2015-04 database with a structure that is representative of the global dataset in order not to bias the results.

The experiments were performed on a server equipped with an Intel(R) Xeon(R) CPU E5-2637 v2 processor at 3.50GHz clock speed, with 172 GB of RAM, 1 TB of disk space running under the Ubuntu 18.04.2 LTS 64-bit operating system.

5.2 Experiments

We ran our algorithm 10 times with populations equal to 100, 200, and 500, respectively, with the idea of comparing the results and observing which parameters are optimal in order to obtain credible axioms.

The parameters chosen, particularly in terms of populations and total effort, were reduced compared to the studies performed by [3–5]. Indeed, the observed execution times are very important. We observe an average time of execution for a population of 100 of about 1 hour, 1 hour and a half on average for an execution with a population of 200 and 6 hours and a half on average for a population with 500 individuals.

The tool allows us to evaluate the axioms on the training dataset for each run, in particular to save time on the evaluation of poor quality axioms. In addition, we can specify, during one or more generations, the evaluation of the axioms on the complete dataset as a complement, in order to confirm or not the accuracy of the axioms. The experiments provided us with a collection of axioms

evaluated on the training dataset and on the full DBPedia 2015-04 dataset after 30 generations of our algorithm, thus providing us with two distinct sets of measures that we can observe, notably in the table 2.

Table 2: Number of complex axioms for which we observe a non-zero possibility after 10 executions

Parameter	Number of axioms	
	with training dataset	with full dataset
100	476	358
200	525	525
500	1911	1888

We notice that a large part of the axioms found and evaluated from our training dataset, keep a non-zero possibility in our full dataset, which confirms the suitability of this approach.

Let us take a deeper look at a case of axiom found with our algorithm. For example, the following axiom:

```
SubClassOf (
    ObjectSomeValuesFrom (
        <http://dbpedia.org/property/leagueTopscorer>
        <http://dbpedia.org/ontology/Agent>
    )
    ObjectSomeValuesFrom (
        <http://dbpedia.org/ontology/league>
        <http://dbpedia.org/ontology/Agent>
    )
)
```

has a possibility of **0.649** with the training dataset, so we find a fairly low rate of counterexamples, which leads us to believe that it would be quite likely to be true. To this end, we also evaluate this individual with the full DBPedia dataset, which contains 9,170,623 classes and is thus better provided with information to give us a more representative evaluation. Given the time limit and the size of the database, our queries reach the imposed time limit because the server does not have the time to process them, it would take a considerable amount of time to be able to evaluate each axiom in a long way but this does not suit our approach. For this reason, we have evaluated this example without time limitation. We found **62 confirmations** for this axiom against **1784 counter-examples** on the 6008 individuals being concerned by this axiom, we thus obtain a possibility of **0.289** and a total evaluation time of **just over one hour**. These results suggest that we should not consider this axiom.

We also found two axioms that caught our attention due to the fact that they had a non-zero necessity (and therefore a possibility equal to 1):

```
(1)
SubClassOf (
    ObjectSomeValuesFrom (
        <http://dbpedia.org/property/narrated>
        <http://dbpedia.org/ontology/Agent>
    )
    <http://dbpedia.org/ontology/Work>
)
(2)
SubClassOf (
    ObjectSomeValuesFrom (
        <http://dbpedia.org/property/parentPeak>
        <http://dbpedia.org/ontology/Place>
    )
    <http://dbpedia.org/ontology/Park>
)
```

The different axioms have been tested, in the same way, without time limitation on the complete database. We have in the case (1), **1052 confirmations** against only **5 counterexamples** on the 1283 individuals concerned, giving a very strong possibility: **0.912** completed in **47 minutes**, we can say that this axiom is quite possible. In case (2), we found **25 confirmations** against **1322 counterexamples** on the 2352 individuals concerned. Thus, this axiom is, in terms of possibility, very weak: **0.101** completed in **53 minutes**.

We obtain quite good performances regarding the global evaluation with our evolutionary algorithm, where we see that the curves are quite high overall over 30 generations with notable variations and some rare runs where the fitness is quite low, hence the interest to run the algorithm several times in order to obtain a set of suitable results. It should be noted that runtime errors are quite rare and are mainly due to the Virtuoso server which, from time to time, tends to crash.

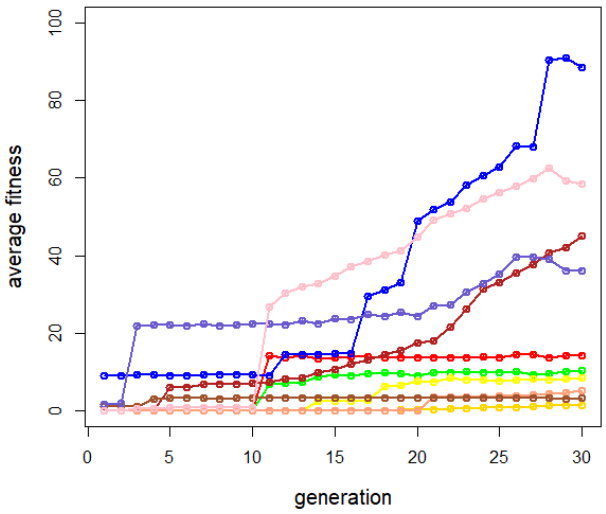

Figure 3: Evolution of average fitness over 10 executions with the same parameters (those that have been successfully completed) for a population of 100 axioms.

6 CONCLUSION

We have seen through our development and the implementation of our method, punctuated by the results obtained, that our method

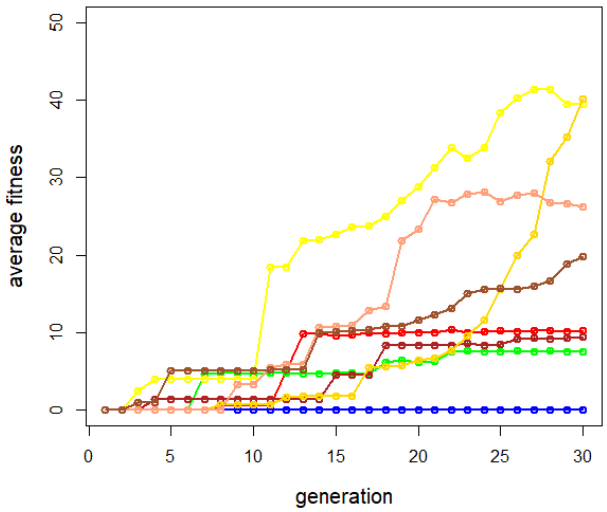

Figure 4: Evolution of average fitness over 10 executions with the same parameters (those that have been successfully completed) for a population of 200 axioms.

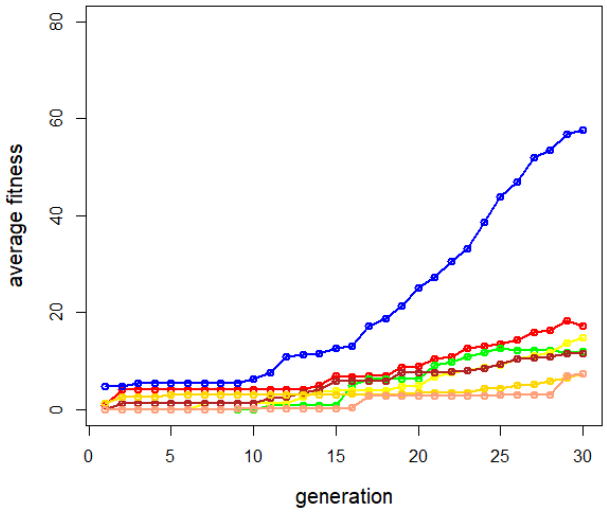

Figure 5: Evolution of average fitness over 10 executions with the same parameters (those that have been successfully completed) for a population of 500 axioms.

allows us to obtain subsumption axioms composed of complex classes that are quite relevant by avoiding some of the difficulties related to the discovery of these axioms, especially in the evaluation. We can hypothesize that higher parameters and a higher evaluation time limit could allow us to obtain better results. To this end, we will carry out optimization work in the different processes of our approach: one of our first lines of thought is to parallelize the evaluation of the different axioms. Indeed, the evaluation of axioms is done in a procedural way, but there is no dependence between each axiom, which allows us to parallelize the evaluation of several axioms at the same time. This track can allow us to save a lot of execution time. We plan to extend our search for axioms composed of complex classes to other types of axioms among those provided by OWL 2.

ACKNOWLEDGMENTS

This research was carried out in the Wimmics team, which is a joint research team of Université Côte d'Azur, Inria, and I3S. Our research motto: AI in bridging social semantics and formal semantics on the Web.

This work has been partially supported by the French government, through the 3IA Côte d'Azur "Investments in the Future" project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

REFERENCES

[1] Patel-Schneider P. Grau B.C., Motik B. 2012. *OWL 2 Web Ontology Language direct semantics (second edition).* W3C Recommendation. W3C.
[2] Staab S. Guarino N., Oberle D. 2009. What Is an Ontology ? In *Handbook on Ontologies.* Springer, 1–17. https://doi.org/10.1007/978-3-540-92673-3_0
[3] Tettamanzi A.G.B. Huong T.H. 2019. An Evolutionary Approach to Class Disjointness Axiom Discovery. IEEE/WIC/ACM International Conference on Web Intelligence, Thessaloniki, Greece, 68–75. https://doi.org/10.1145/3350546.3352502
[4] Tettamanzi A.G.B. Huong T.H. 2020. Grammatical Evolution to Mine OWL Disjointness Axioms Involving Complex Concept Expressions. In *CEC 2020.* IEEE Congress on Evolutionary Computation, Glasgow, 1–8. https://doi.org/10.1109/CEC48606.2020.9185681
[5] Tettamanzi A.G.B. Huong T.H. 2020. Using Grammar-Based Genetic Programming for Mining Disjointness Axioms Involving Complex Class Expressions. In *Ontologies and Concepts in Mind and Machine.* ICCS, 18–32. https://doi.org/10.1007/978-3-030-57855-8_2
[6] Zadeh L.A. 1978. Fuzzy sets as a basis for a theory of possibility. In *Fuzzy Sets and Systems 1.* 3–28.
[7] Völker J. Lehmann J. 2014. Perspectives on Ontology Learning. In *Studies on the Semantic Web.* Vol. 18. IOS Press.
[8] Andrea G.B. Tettamanzi, Catherine Faron-Zucker, and Fabien Gandon. 2017. Possibilistic testing of OWL axioms against RDF data. *International Journal of Approximate Reasoning* 91 (2017), 114–130. https://doi.org/10.1016/j.ijar.2017.08.012
[9] Andrea G. B. Tettamanzi, Thu Huong Nguyen, and Rémi Felin. [n.d.]. RDFMining project. https://github.com/RemiFELIN/RDFMining.
[10] Faron-Zucker C. Tettamanzi A.G.B. and F.L. Gandon. 2014. Testing OWL axioms against RDF facts: A possibilistic approach. In *EKAW.* Lecture Notes in Computer Science, Vol. 8876. Springer, 519–530. https://doi.org/10.1007/978-3-319-13704-9_39
[11] Faron-Zucker C. Tettamanzi A.G.B. and F.L. Gandon. 2015. Dynamically Time-Capped Possibilistic Testing of SubClassOf Axioms Against RDF Data to Enrich Schemas. In *K-CAP (Proceedings of the 8th International Conference on Knowledge Capture, 7)*, Ken Barker and José Manuel Gómez-Pérez (Eds.). Palisades, NY, United States. https://doi.org/10.1145/2815833.2815835
[12] Nguyen T.H. 2021. *Mining the Semantic Web for OWL Axioms.* Ph.D. Dissertation. Université Côte d'Azur, Biot.