EVALUATION AND ANALYSIS OF SHACL SUPPORT BY VALIDATION TOOLS

Università degli Studi di Firenze
Corso di Laurea Magistrale in Ingegneria Informatica
Insegnamento di Security and Knowledge Management

Cosimo Giani

A.A. 2020 - 2021

Contents

1 Introduction

This paper is conceived as a study and analysis of some implementations to evaluate the current support of SHACL.

In recent years there has been a considerable increase in the amount of linked data on the web, which has led large companies, i.e. Google, to the use of semantic data structures and knowledge graphs to improve their services for the user.

What has been said can be interpreted as a sign of the increasing importance of the Semantic Web e with the increase in utilization of graph-based data on the web the need for validation of this type of data has increased as well. For this reason, the W3C consortium has provided a standard called **Sh**ape **C**onstraint **L**anguage (**SHACL**), a specific for validating graph-based data against a set of conditions.

A SHACL validation engine takes as input a data graph and a graph containing shapes declarations and produces a validation report that can be consulted.

As more people work with data coming from a variety of sources, especially for data integration projects, SHACL gives them a way to describe the "shapes" of the data that they're working with so that applications can take better advantage of that data. In addition to describing which properties go with which classes, SHACL lets define constraints on data that, when used by applications, can make it easier to improve the quality of the data with standardized models instead of procedural code.

With this in mind, the adoption of SHACL may influence the future of linked data.

2 Dataset

As for the construction of the dataset used for the evaluation of the SHACL support, *DBpedia* has been used, a project born in 2007 with the aim of extracting structured information from Wikipedia and publishing it on the Web as Linked Open Data in RDF format.
Specifically, through the databus provided by DBpedia itself has been taken a part of the **DBpedia Latest Core Release**, a small subset of the total DBpedia releases which is the tiniest knowledge graph and contains actual data from articles and infoboxes of the English Wikipedia Language Edition (WPLE), with approximately 900 million triples in continuous growth and enriched with rdf:type statements to several ontologies.
The latter fact influenced in particular the construction of the shapes, as the presence of multiple domains can lead to greater diversification and an increase in the complexity of the structure of the shapes themselves.
For evaluation purposes, several subsets of different sizes of this knowledge graph have been created, which, combined with the variety of constructable shapes just mentioned, contributes for different subsets of the DBpedia Latest Core Release or other datasets to significantly vary the number of validated classes and properties.
In particular this can also lead to an increase in violations, since the SHACL shapes used in this project were explicitly designed for the dataset constructed in the manner just described.

3 Shapes construction

In order to validate data, SHACL requires two different inputs. First, a data graph has to be available that may contain invalid or malformed entries: this data graph is represented by the constructed dataset described in chapter 2. The second required input is the shapes graph that declares constraints upon the given data from the data graph.

There exist two types of shapes: node shapes and property shapes. A *node shape* specifies constraints that need to be fulfilled by focus nodes. A focus node is an RDF term from the data graph that is validated against a shape. This means that a node shape specifies a focus node for which the remaining constraints of the shape has to apply. A *property shape* specifies constraints on nodes that are reachable from a given focus node by following a property path. This allows to define clear restrictions on possible values for subjects of a triple or quad, e.g. cardinality constraints, value types and value ranges. For this work, in order to obtain an appropriate and meaningful evaluation, the shapes were therefore created and identified following the main occurrences for each class that constitutes the DBpedia dataset, as shown here:

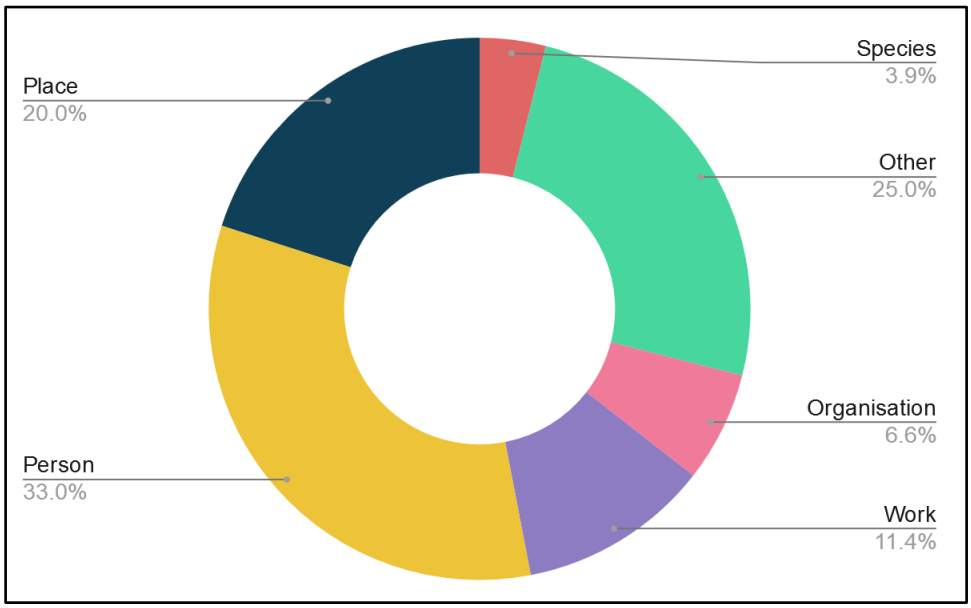

Figure 1: instances per class - DBpedia.

In addition to the basic classes listed above (except for the ***Other*** class), some shapes of sub-classes have been introduced in order to diversify the

results produced and make them more interesting, i.e. under ***Place*** was introduced the shape ***Airport***, or under ***Work*** the shapes ***Film*** and ***Song*** have been chosen. From this perspective, for the aforementioned ***Other*** class, ***Events*** of different natures were considered, such as *elections*, *sports*, *military conflicts*.
This was made for a total of **14** shapes, which were used for validation during the testing phase.
Apart from the instrumental shape-based constraint components, the shapes include the cardinality constraint components *sh:maxCount* and *sh:minCount*, value type constraint components like *sh:datatype*, and string-based constraint components like *sh:pattern*.
It was also used the logical constraint component *sh:or*, which led to different results in the validation reports depending on the graph database.

4 Evaluation

This chapter will discuss the tools used for the SHACL validation of the dataset in question, in terms of ease of use and any difficulties encountered in their use.

4.1 Validation tools

Regarding the selection of tools to perform SHACL validation, this was not a simple operation, as, at the time of writing, the SHACL support was not very widespread and lacking sufficient documentation.
In this project, the following three tools were selected and used for the validation:

- **TopBraid** by TopQuadrant
- **RDF4J** by Eclipse Foundation
- **Neosemantics** (Neo4J) by Neo Technology

TopBraid SHACL API is an open source implementation of the W3C Shapes Constraint Language (SHACL) based on *Apache Jena*, that offers the component TDB, which acts as an RDF store and query database. It can be used to perform SHACL constraint checking and rule inferencing in any Jena-based Java application.
RDF4J is an open-source Java framework and part of the *Eclipse* project. The functionality of the framework is focused on storing, querying, and analysing RDF data. In this case for the RDF store the in memory RDF repository called *SailRepository* has been selected, according to the documentation.
Neo4J is an open-source graph database management system and developed entirely in Java. It is a fully transactional database, which is integrated into the applications allowing them to operate in stand alone manner. In particular, this provides a plugin called **Neosemantics** (n10s), which enables the use of RDF and its associated vocabularies in Neo4j and which can be used to validate the used graph against constraints expressed in SHACL.

4.2 Implementation details

The main criterion on which the tests were based and subsequently the necessary conclusions were drawn is measuring the time that it takes to generate a SHACL report, therefore the time required to write the aforementioned report will not be taken into account in the measurements. For the type of temporal evaluation chosen, to avoid that the Java implementations of the two frameworks used could influence the final results, these were kept relatively simple.
In addition information regarding the difficulty of using SHACL, the usefulness of the documentation and the quality of the SHACL report produced are provided.

4.2.1 TopBraid framework

TopBraid is a solution based on the Apache Jena framework that was used in its form of API to be able to obtain a SHACL validation of data contained and stored within a triplestore, which in this case was implemented through **Jena TDB**, a component of Jena for RDF storage and query.
To measure the time of validating data stored in the provided triplestore (TDB) a Java application has been written: after having loaded the aforementioned data and having read the shapes, the SHACL validation takes place by a method belonging to the **ValidationUtil** class and provided by the framework itself, named **validationModel**, which receives in particular the data model and the shapes model as parameters and performs the validation, returning subsequently an instance of **sh:ValidationReport** as result. A final check takes care to determine whether this report model complies with the SHACL standard or not: if so, a report file is written, which allows the user to be able to evaluate and verify violations.
However, this phase encountered some adversities, as the documentation was lackluster and only provided an example for validating datasets provided as files.

4.2.2 RDF4J framework

In this approach to validate data from the provided triplestore, implementing an application that uses the RDF4J framework is required.

The triplestore used is called **SailRepository**, a repository that operates directly on top of a **Sail** - that is, a particular database. This is the class most commonly used when accessing/creating a local RDF4J repository.
The application connects to the *SailRepository*, loads the SHACL shapes from a file in a transactional manner and performs the validation.
The latter is in particular achieved through the **commit()** of the data that are loaded into the triplestore: if a violation of a SHACL constraint occurs, a **RepositoryException** is thrown after the entire data is checked. From this exception the validation report can be generated and it is written to a file.
For this case study, on the contrary, the documentation was perceived as rather exhaustive, even providing a complete working example and listing in detail which SHACL features are supported by the framework. Such a thing was particularly helpful during the experimental phase, as described in detail in the next chapter.

4.2.3 Neosemantics

Neosemantics is a plugin of the well-known Neo4J database management system that enables the use of RDF and its associated vocabularies e runs as an extension to the Neo4j database.
Neo4J comes as a desktop application and regarding the plugin installation some preliminary configuration steps were required in order to make it work. Particularly, it is necessary to create a local database for data storage, which, at the time of writing this paper, does not support SHACL validation in its most recent version, that is 4.3.* (4.3.0 or advanced releases). For this reason **version 4.2.*** was used - in this case every version is accepted.
The ability to perform SHACL validation takes place in two ways.
The first one consists of the classic UI made available by Neo4J itself called **Neo4J Browser**, which allows to execute specific commands provided by the plugin and written in **Chyper**, a graph query language that allows users to store and retrieve data from the graph database. Through these commands the aforementioned validation is performed and a complaint report is produced, which can be downloadable as a table or as a JSON file.
The alternative consists of a real GUI (or *GraphApp*) of the plugin, easier to use than the interface just described, which allows to carry out the task of this elaborate without further difficulties by simply following in an

intuitive way the steps reported by the interface and organizes the information regarding the violations in a table.
Although the use of both the aforementioned interfaces is not too complicated, in order to be able to exploit the features of Neosemantics it was necessary to fulfill additional work, since the documentation for the installation of this "young" plugin was perceived to be not quite exhaustive.

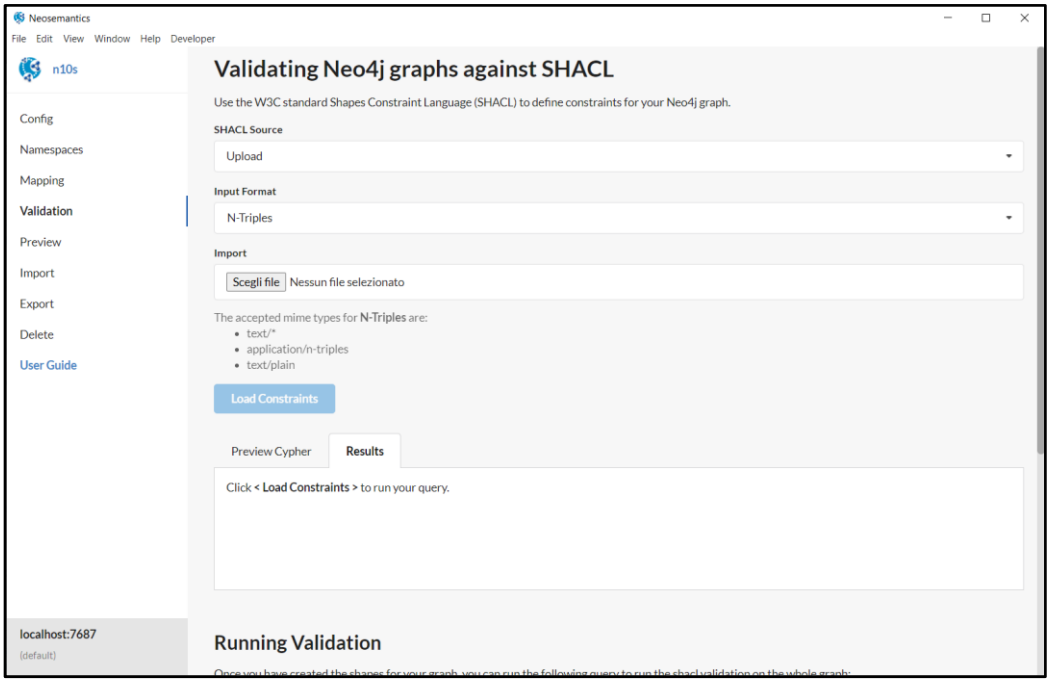

Figure 2: *Neosemantics* graphical user interface.

5 Tests and results

Following the preliminary phase during which the validation tools were implemented and studied, a series of tests were performed to determine the quality of the SHACL support.
The first experiments conducted consist of speed-oriented tests and related memory use. Furthermore, to make the evaluation more accurate, the support of SHACL features, i.e. constraint components, was analyzed, which led to further considerations.
This experimental phase took place on a machine with an Intel Core i7-8750H @ 2.20GHz processor and 16GB of RAM.

5.1 Average time

These experiments involved the use of the dataset constructed as reported in chapter 2 by calculating the average execution time to detect violations and any amount of memory required.
To make the experimental scenario more likely, the tests were performed with increasing fragments of the dataset: specifically, subsets composed of 1000, 10.000, 100.000, 1M and 5M triples were used, in order to read the possible trend of the tool execution time as the dataset size increases.
To obtain the average performance, 10 measurements were carried out, but in order to reduce the effect of disk caching, these tests were distributed and not repeated sequentially.
The graphs produced by these operations with the related measurements are presented in appendix A at the end of the paper.
Below in figure 3 is only presented the comparison chart of the average performance of the tools considered. By studying its trend it is possible to compare the times of all the tools and it seems quite clear that in terms of validation speed Neosemantics is the most performing, followed by RDF4J and finally TopBraid.
By observing the initial part of the graphs, the stretch that reaches up to 1M of triples, it might seem that TopBraid is clearly inferior to the other two.
For this reason, as a precaution, tests up to 5M of triples were also conducted, leading the tool into question under a new perspective: in fact, by placing attention on the central part (1M - 5M) of the graph, it is possible to

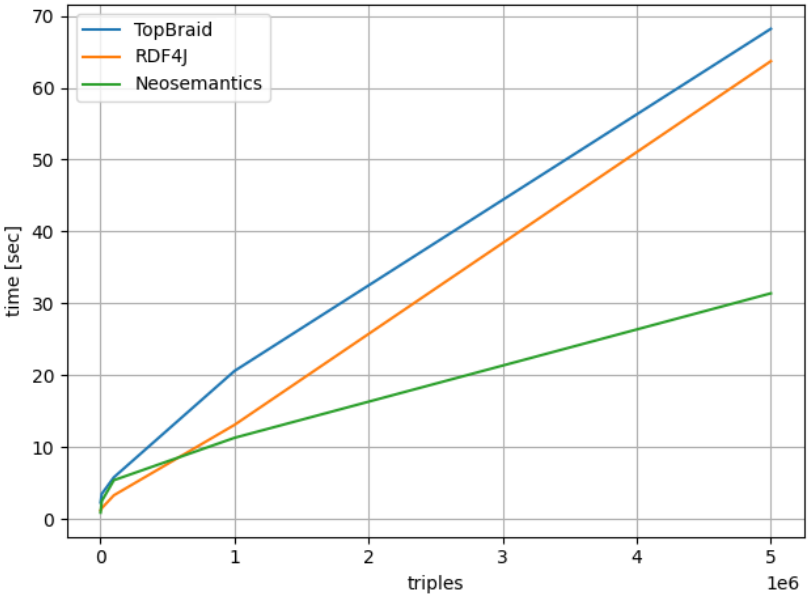

Figure 3: comparison between average validation times.

see how the gap between TopBraid and RDF4J decreases as the number of data increases and, vice versa, the gap with Neosemantics gets wider.
All this seems to suggest that with a possible larger dataset the TopBraid framework could make this distance null and even surpass RDF4J in terms of speed.
This last deduction is further strengthened if the attention is paid to the following graph, which shows the average memory required by each tool (to read the related measurements see appendix A):

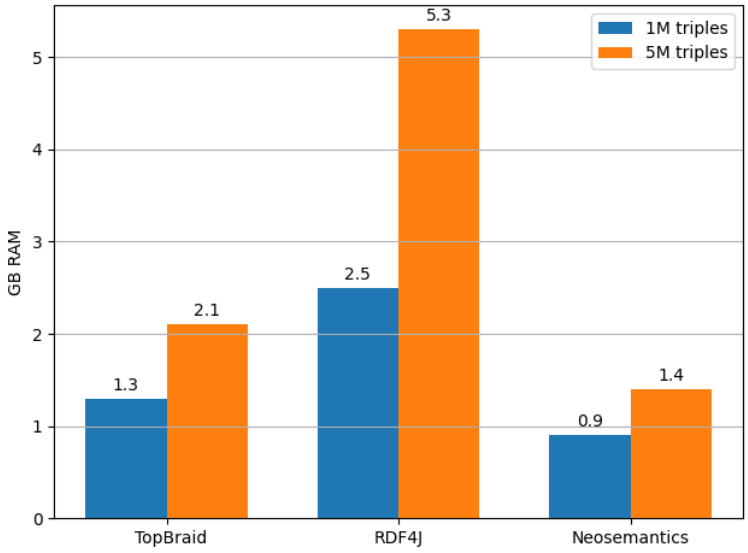

Figure 4: average amount of memory used by the tools with 1M and 5M triples.

As can be seen from this graph, Neosemantics is the one that manages memory in the most efficient way.
Regarding the two frameworks, however, it is noted that RDF4J is the most demanding one, since as the number of triples it works with increases, it becomes necessary to double the amount of memory used, while TopBraid does not significantly increase its needs.
Therefore, for what has just been highlighted, the previously introduced hypothesis of a possible worsening in the validation speed for larger datasets seems quite plausible, precisely due to the inability of RDF4J to manage its resources efficiently.

5.2 Feature support

Finally some experiments have been conducted with the aim to analyze the support of SHACL features, in order to further compare the tools used during the validation.
In other words, the tests consisted of guided violations to verify the actual ability of these tools in detecting and managing the SHACL features.
This obviously required to write some extra shapes, as those built in chapter 3 were not suitable for the purpose. In a similar way, small fragments of erroneous data were collected or constructed.
This section therefore presents the main features and few examples of possible related violations. All the generated violations have been reported in appendix B.

5.2.1 Value type constraint components

These constraint components can be used to restric the type of value nodes.

- **sh:datatype**
  It specifies a condition to be satisfied with regards to the datatype of each value node. The values of *sh:datatype* are typically datatypes, such as *xsd:string*. All the tools have detected this elementary constraint.

```
[
  {
    "focusNode": "http://dbpedia.org/resource/Esselunga",
    "nodeType": "http://dbpedia.org/ontology/Organisation",
    "shapeId": "bnode://id/node1fe6rogu8x43",
    "propertyShape": "http://www.w3.org/ns/shacl#DatatypeConstraintComponent",
    "offendingValue": 23.094,
    "resultPath": "http://dbpedia.org/property/numEmployees",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": "property value should be of type http://www.w3.org/2001/XMLSchema#integer"
  }
]
```


Figure 5: *sh:datatype* violation example generated by Neosemantics - JSON file. An integer value was expected, but the data type is instead double.

- **sh:nodeKind**

  It specifies a condition to be satisfied by the RDF node kind of each value node. A shape has at most one value for *sh:nodeKind*. The tools were able to make use of this feature as well.


```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                               sh:ValidationResult ;
    sh:focusNode                    <http://dbpedia.org/resource/Ordinary_Happiness> ;
    sh:resultMessage                "Film title is not an IRI or is invalid" ;
    sh:resultPath                   <http://dbpedia.org/property/title> ;
    sh:resultSeverity               sh:Violation ;
    sh:sourceConstraintComponent    sh:NodeKindConstraintComponent ;
    sh:sourceShape                  [] ;
    sh:value                        "Ordinary Happiness"@en
] ;
```


Figure 6: *sh:nodeKind* violation example generated by TopBraid. The node kind was expected to be IRI, but the value passed is Literal.

5.2.2 Cardinality constraint components

The following constraint components represent constrictions on the number of value nodes for the given focus node.

- **sh:minCount - sh:maxCount**

  These constraint components represent restrictions on the number of value nodes for the given focus node. The first one specifies the minimum number of value nodes that satisfy the condition. If the minimum cardinality value is 0 then this constraint is always satisfied and so may be omitted. The second one specifies the maximum number of value nodes that satisfy the condition. Like the previous ones, it is part of that set of features that are to be considered basic for a tool which is aiming at SHACL support.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:3636add0-70ac-4f7b-9585-3a4d10bd87dd a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Chōfu_Airport>;
  sh:resultPath <http://dbpedia.org/property/iata>;
  sh:sourceConstraintComponent sh:MinCountConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fe6quvgvx52 .

_:node1fe6quvgvx52 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/iata>;
  sh:minCount 1 .
```

Figure 7: *sh:minCount* violation example generated by RDF4J. The IATA code is considered necessary for an airport, but the Chofu Airport page is missing this value.

5.2.3 Value range constraint components

- **sh:minInclusive - sh:maxInclusive and sh:minExclusive - sh:maxExclusive**

  These constraint components specify value range conditions to be satisfied by value nodes that are comparable via operators such as <, <=, > and >=. Even in this case this type of constraint is detected and supported by every tool.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                              sh:ValidationResult ;
    sh:focusNode                   <http://dbpedia.org/resource/Gorgan_Airport> ;
    sh:resultMessage               "Place latitude not in standard range" ;
    sh:resultPath                  <http://www.w3.org/2003/01/geo/wgs84_pos#lat> ;
    sh:resultSeverity              sh:Violation ;
    sh:sourceConstraintComponent   sh:MinInclusiveConstraintComponent ;
    sh:sourceShape                 [] ;
    sh:value                       "-100.0000"^^<http://www.w3.org/2001/XMLSchema#float>
] ;
```

Figure 8: *sh:minInclusive* violation example generated by TopBraid. The standard states that the range of the latitude coordinates is limited between -90 (South) to 90 (North). In this fictional example the corresponding value is -100, hence the violation.

5.2.4 String-based contraint components

The constraint components in this section have in common that they specify conditions on the string representation of value nodes.

- **sh:minLength - sh:maxLength**

  The first specifies the minimum string length of each value node that satisfies the condition. The latter, on the other hand, specifies the

maximum string length of each value node that satisfies the condition. The violations of these constraints were detected by every tool.


```
[
    {
    "focusNode": "http://dbpedia.org/resource/Gorgan_Airport",
    "nodeType": "http://dbpedia.org/ontology/Airport",
    "shapeId": "bnode://id/node1ffn1p6a7x2",
    "propertyShape": "http://www.w3.org/ns/shacl#MaxLengthConstraintComponent",
    "offendingValue": "OINGX",
    "resultPath": "http://dbpedia.org/property/icao",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": ""
  }
]
```


Figure 9: *sh:maxLength* violation example generated by Neosemantics - JSON file. The standard for an ICAO code length is 4 characters, but for the validation purpose the one in the example was extended to 5 characters.

- **sh:pattern**

  It specifies a regular expression that each value node matches to satisfy the condition. All the tools in this paper are able to support this elementary constraint.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                               sh:ValidationResult ;
    sh:focusNode                    <http://dbpedia.org/resource/José_Enrique_Varela> ;
    sh:resultMessage                "Person birth date has invalid format" ;
    sh:resultPath                   <http://dbpedia.org/property/birthDate> ;
    sh:resultSeverity               sh:Violation ;
    sh:sourceConstraintComponent    sh:PatternConstraintComponent ;
    sh:sourceShape                  [] ;
    sh:value                        "1891-1-1"^^<http://www.w3.org/2001/XMLSchema#date>
] ;
```

Figure 10: *sh:pattern* violation example generated by TopBraid. The birth date pattern should be Y-M-D with two digits for both month and day, as stated by the standard. In this fictional example the date pattern was changed with only one digit for the month and one for the day, hence the violation.

- **sh:languageIn**

  The condition specified by sh:languageIn is that the allowed language tags for each value node are limited by a given list of language tags. For this case, unlike the previous ones, the support of Neosemantics is not effective, since it is not able to perceive the violation of a constraint of this type.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:1ac39aca-1b2c-4a4b-aae1-73e4c357bb06 a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Clash_by_Night>;
  sh:value "Clash by Night"@en;
  sh:resultPath <http://dbpedia.org/property/title>;
  sh:sourceConstraintComponent sh:LanguageInConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ffn2gcomx1 .

_:node1ffn2gcomx1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/title>;
  sh:languageIn _:node1ffn2gcomx2 .

_:node1ffn2gcomx2 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "it";
```

Figure 11: *sh:languageIn* violation example generated by RDF4J. For this example the language requested for the film title was "Italian", but the actual language is "English".

5.2.5 Property pair constraint components

The constraint components in this section specify conditions on the sets of value nodes in relation to other properties. These constraint components can only be used by property shapes.

- **sh:equals**

  It specifies the condition that the set of all value nodes is equal to the set of objects of the triples that have the focus node as subject and the value of *sh:equals* as predicate. The experiments conducted for this feature reported that TopBraid is the only one able to work with this constraint.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                              sh:ValidationResult ;
    sh:focusNode                   ex:bob ;
    sh:resultMessage               "Must have same values as ex:firstName" ;
    sh:resultPath                  ex:givenName ;
    sh:resultSeverity              sh:Violation ;
    sh:sourceConstraintComponent   sh:EqualsConstraintComponent ;
    sh:sourceShape                 _:b0 ;
    sh:value                       "Robert"
] ;
```

Figure 12: *sh:equals* violation example generated by TopBraid. The report states that the given name of the person in this fictional example is equal to the first name, hence the violation.

- **sh:disjoint**

  It specifies the condition that the set of value nodes is disjoint with

the set of objects of the triples that have the focus node as subject and the value of *sh:disjoint* as predicate. As for the previous constraint, TopBraid is the only one able to support this feature.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     ex:carol ;
    sh:resultMessage                 "Property must not share any values with ex:lastName" ;
    sh:resultPath                    ex:givenName ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:DisjointConstraintComponent ;
    sh:sourceShape                   [] ;
    sh:value                         "Carol"
] ;
```

Figure 13: *sh:disjoint* violation example generated by TopBraid. Similarly to the previous example, the violation is due to the fact that the given name of the example person is equal (and therefore not different) to the last name.

- **sh:lessThan**

  It specifies the condition that each value node is smaller than all the objects of the triples that have the focus node as subject and the value of *sh:lessThan* as predicate. The only tool that supports this feature is once again TopBraid.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/Walter_Schuck> ;
    sh:resultMessage                 "Person birth date is greater than death date" ;
    sh:resultPath                    <http://dbpedia.org/property/birthDate> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:LessThanConstraintComponent ;
    sh:sourceShape                   [] ;
    sh:value                         "2000-07-30"^^<http://www.w3.org/2001/XMLSchema#date>
] ;
```

Figure 14: *sh:lessThan* violation example generated by TopBraid. In this really fictional example, the person birth date was changed so that it was greater than the death date, i.e. birth chronologically prior to death - obviously impossible - hence the violation.

5.2.6 Logical constraint components

- **sh:or**

  It specifies the condition that each value node conforms to at least one of the provided shapes. This is comparable to disjunction and the logical "or" operator. Following an analysis of the reports produced, it was possible to note that this feature is evaluated differently depending on the tool used.

RDF4J interprets any constraint in the *sh:or* construct, and *sh:or* itself as a violation, if none of the constraints are fulfilled. This leads to an higher number of violations than the TopBraid tool, which, on the other hand, assumes a different behavior and reports one and only violation. Neosemantics however does not support this feature in any way.

The evident difference in the treatment of the constraint or by RDF4J seems to reinforce once again the idea conceived during the previous testing phase: the higher number of violations reported for this case, combined with the mismanagement of resources, can easily convince that, in the eventuality of a dataset with more data and with a number of *sh:or* constraints higher than that studied in this paper, the time required by RDF4J for validation may further deteriorate.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:75848eef-eeb6-4f26-8411-c4ea826d41b6 a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/The_Fatal_Woman>;
  sh:value 1.9E0;
  sh:resultPath <http://dbpedia.org/property/released>;
  sh:sourceConstraintComponent sh:OrConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ff2uefglx1 .

_:node1ff2uefglx1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/released>;
  sh:or _:node1ff2uefglx2 .

_:node1ff2uefglx3 a sh:NodeShape;
  sh:datatype <http://www.w3.org/2001/XMLSchema#date>;
  sh:pattern "^\\d{4}-\\d{2}-\\d{2}$" .

_:node1ff2uefglx5 a sh:NodeShape;
  sh:datatype <http://www.w3.org/2001/XMLSchema#integer>;
  sh:pattern "^\\d{4}" .
```

Figure 15: *sh:or* violation example generated by RDF4J. Since for the example film the *released* property does not have a correct value, all the elements of the *sh:or* list produce an error.

- **sh:and**

  It specifies the condition that each value node conforms to all provided shapes. This is comparable to conjunction and the logical "and" operator. In other words, a node conforms to a shape containing the

sh:and operator if it conforms to all the shapes linked by it. This feature is only supported by TopBraid and RDF4J at the moment. In particular, as for the *sh:or* case, RDF4J generates a violation for each of the elements of the *sh:and* list, even if this time it is sufficient to violate only one constraint. TopBraid similarly to before produces a single validation result in case of a violation.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:32b009cd-a2e9-4126-8dfe-df0e0a55b76e a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Ordinary_Happiness>;
  sh:value "Ordinary Happiness"@en;
  sh:resultPath <http://dbpedia.org/property/title>;
  sh:sourceConstraintComponent sh:AndConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ffkumfsmx1 .

_:node1ffkumfsmx1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/title>;
  sh:and _:node1ffkumfsmx2 .

_:node1ffkumfsmx3 a sh:NodeShape;
  sh:nodeKind sh:Literal .

_:node1ffkumfsmx5 a sh:NodeShape;
  sh:languageIn _:node1ffkumfsmx6 .

_:node1ffkumfsmx6 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "it";
```

Figure 16: *sh:and* violation example generated by RDF4J. After detecting an error in the film title, in the report are included as many violations as the components of the *sh:and* constraint.

5.2.7 Shape-based constraint components

The constraint components in this section can be used to specify complex conditions by validating the value nodes against certain shapes.

- **sh:qualifiedValueShape and sh:qualifiedMinCount - sh:qualifiedMaxCount**

  It specifies the condition that a specified number of value nodes conforms to the given shape. Each *sh:qualifiedValueShape* can have: one value for *sh:qualifiedMinCount*, one value for *sh:qualifiedMaxCount* or, one value for each, at the same subject. Qualified value shapes declare that a specified number of nodes conform to some shape. The shape is declared by the *sh:qualifiedValueShape* parameter and the parameters *sh:qualifiedMinCount* and *sh:qualifiedMaxCount* declare

the minimum and maximum number of values of that shape. A typical use case for qualified value shapes is to model repeated properties whose values must conform to different shapes. Again, the only tool not able to work with these features is Neosemantics.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     ex:dave ;
    sh:resultMessage                 "Less than 1 values for the qualified shape" ;
    sh:resultPath                    ex:parent ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:QualifiedMinCountConstraintComponent ;
    sh:sourceShape                   [] ;
] ;
```

Figure 17: *sh:qualifiedMinCount* violation example generated by TopBraid. For this case, a shape has been created, which indicates that each person must have at least one male and one female parent. The person in this fictitious example has no information on the gender of either parent, hence the violation.

- **sh:qualifiedValueShapesDisjoint**

  This is not technically a feature, but an optional parameter of *sh:QualifiedMinCountConstraintComponent* and *sh:QualifiedMaxCountConstraintComponent*. If set to true then (for the counting) the value nodes must not conform to any of the sibling shapes. Despite this, a special experiment was conducted for this constraint, in orderd to test the actual ability to support the feature and it was found that, like the previous constraints of this section, the only tool unable to validate this constraint is Neosemantics.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     ex:hand ;
    sh:resultMessage                 "Less than 1 values, not well-formed thumb" ;
    sh:resultPath                    ex:digit ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:QualifiedMinCountConstraintComponent ;
    sh:sourceShape                   [] ;
] ;
```

Figure 18: *sh:qualifiedValueShapesDisjoint* violation example generated by TopBraid. For this case has been stated that a hand, which has at most 5 properties/digits, has exactly one of them that is an instance of *ex:Thumb* while exactly 4 of them are an instance of *ex:Finger*, but thumbs and fingers must be disjoint. For the example hand, this does not have a well-formed thumb by its definition, i.e. it has a digit which is simultaneously a thumb and a finger, therefore it is not treated as a thumb, hence the violation.

The current support provided by the aforementioned SHACL validation tools, in relation to the features just discussed, can be summarized in the following table:

Feature	TOPBRAID	RDF4J	NEOSEMANTICS
DATATYPE	✓	✓	✓
NODEKIND	✓	✓	✓
MIN/MAX COUNT	✓	✓	✓
MIN/MAX INCLUSIVE	✓	✓	✓
MIN/MAX EXCLUSIVE	✓	✓	✓
MIN/MAX LENGTH	✓	✓	✓
PATTERN	✓	✓	✓
LANGUAGE IN	✓	✓	✗
EQUALS	✓	✗	✗
DISJOINT	✓	✗	✗
LESS THAN	✓	✗	✗
AND	✓	✓	✗
OR	✓	✓	✗
QUALIFIED VALUE SHAPE	✓	✓	✗
QUALIFIED MIN/MAX COUNT	✓	✓	✗
QUALIFIED VALUE SHAPE DISJOINT	✓	✓	✗

Figure 19: summary table of SHACL feature support.

5.3 Report comparison

As a final point, some differences are highlighted between the reports produced by each tool. For this purely demonstrative purpose, take the following property path as a reference:

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix dbo: <http://dbpedia.org/ontology/> .
@prefix dbp: <http://dbpedia.org/property/> .
@prefix : <http://SEKM_EXAM.com/ns#> .

:OrganisationShape a sh:NodeShape;
    sh:targetClass dbo:Organisation;
    sh:property [
        sh:path dbp:numEmployees;
        sh:datatype xsd:integer;
        sh:message "Organisation number of employees has invalid data type";
    ].
```

Figure 20: property path for the number of employees of an organisation. The organisation considered for this example is *Esselunga*.

The reports generated by the tools as a result of a violation are shown below:

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                               sh:ValidationResult ;
    sh:focusNode                    <http://dbpedia.org/resource/Esselunga> ;
    sh:resultMessage                "Organisation number of employees has invalid data type" ;
    sh:resultPath                   <http://dbpedia.org/property/numEmployees> ;
    sh:resultSeverity               sh:Violation ;
    sh:sourceConstraintComponent    sh:DatatypeConstraintComponent ;
    sh:sourceShape                  [] ;
    sh:value                        "23.094"^^<http://www.w3.org/2001/XMLSchema#double>
] ;
```

Figure 21: validation report generated by *TopBraid* for the reference example.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:node1fe97asp7x942 sh:result _:fefb7230-168c-4123-82f2-1b5026a34f97.

_:fefb7230-168c-4123-82f2-1b5026a34f97 a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Esselunga>;
  sh:value 2.3094E1;
  sh:resultPath <http://dbpedia.org/property/numEmployees>;
  sh:sourceConstraintComponent sh:DatatypeConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fe97at73x43 .

_:node1fe97at73x43 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/numEmployees>;
  sh:datatype <http://www.w3.org/2001/XMLSchema#integer> .
```

Figure 22: validation report generated by *RDF4J* for the reference example.

```
[
  {
    "focusNode": "http://dbpedia.org/resource/Esselunga",
    "nodeType": "http://dbpedia.org/ontology/Organisation",
    "shapeId": "bnode://id/node1fe6rogu8x43",
    "propertyShape": "http://www.w3.org/ns/shacl#DatatypeConstraintComponent",
    "offendingValue": 23.094,
    "resultPath": "http://dbpedia.org/property/numEmployees",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": "property value should be of type http://www.w3.org/2001/XMLSchema#integer"
  }
]
```


Figure 23: validation report generated by *Neosemantics* - JSON file - for the reference example.

focusNode	nodeType	shapeId
http://dbpedia.org/resource/Esselunga	http://dbpedia.org/ontology/Organisation	bnode://id/node1fb6qj3r9x43

propertyShape	offendingValue	resultPath
http://www.w3.org/ns/shacl#DatatypeConstraintComponent	23.094	http://dbpedia.org/property/numEmployees

severity	resultMessage
http://www.w3.org/ns/shacl#Violation	property value should be of type http://www.w3.org/2001/XMLSchema#integer

Figure 24: validation report generated by *Neosemantics* - table view - for the reference example.

By observing what has just been reported, it is possible to notice how the two frameworks are perfectly capable of generating a report compliant with the standard dictated by the W3C, even if, to be absolutely precise, the one produced by RDF4J does not include the *sh:resultMessage* component, which, in some cases, could make the identification of the cause of the violation more difficult.

Neosemantics, on the other hand, does not produce an actual report that follows the characteristics of the SHACL standard, but despite this, the structure given to the information allows to understand the nature of the violations in a very intuitive way.

6 Conclusions and future works

From the information collected and the analysis of the results it is possible to deduce that, in terms of data scalability and execution speed, the Neosemantics tool is the most advantageous, but in relation to the quality of the reports it can be easily agreed that the two frameworks behave in a better way. Furthermore, this apparent advantage could be completely related to the obvious inability of the tool to provide adequate support.
This is to be found in the fact that the plugin is relatively recent and constantly updated by the developers, therefore many of the features have not yet been implemented and thus not supported.
SHACL is currently still very new as a de-facto standard, ideed, after years, an extremely small number of graph databases offering implementations still persist. This, combined with the limited documentation provided by the few tools that support this type of validation, is slowing the growth in adoption of this standard.
The results and considerations made in the previous chapter, however, allow to say that the choice of one tool over another by a possible user is to be weighed considering the actual needs that one has, in addition to the more or less high availability of resources of the machines hosting these graph databases.
If the work to be conducted is not too demanding in terms of shapes, or particularly complex specific features are not necessary, Neosemantics offers basic support and satisfactory validation times. On the other hand, if there is a real need to refine the shapes in order to make them more restrictive, the frameworks are undoubtedly the best choice, as they offer more complete support for SHACL features.
As a continuation of this work, one possible approach could be to work not with a subset, but with the dataset in its entirety, in order to achieve a more accurate and representative performance evaluation, as previously highlighted in the case of RDF4J.
Furthermore, it might be interesting to try SHACL validation on different datasets, in order to further compare what has already been studied and yield the considerations made even more precise, obviously taking into account making appropriate changes to the shapes used.

7 References

- https://www.w3.org/TR/shacl/
- https://www.topquadrant.com/technology/shacl/tutorial/
- https://www.dbpedia.org/resources/ontology/
- https://www.dbpedia.org/resources/latest-core/
- https://rdf4j.org/documentation/programming/shacl/
- https://rdf4j.org/about/
- https://rdf4j.org/documentation/tutorials/maven-eclipse-project/
- https://rdf4j.org/documentation/programming/repository/
- https://github.com/TopQuadrant/shacl
- https://jena.apache.org/documentation/tdb/index.html
- https://jena.apache.org/documentation/tdb/configuration.html
- https://neo4j.com/docs/labs/nsmntx/current/validation/
- https://neo4j.com/labs/neosemantics/
- https://github.com/neo4j-labs/neosemantics
- R. Schaffernrath, D.Proksch, M. Kopp, I. Albisani, O. Panasiuk, A. Fensel, "*Benchmark for Performance Evaluation of SHACL Implementations in Graph Databases*", Rules and Reasoning, pp. 82-96, August 2020.
- M. R. A. Rashid, G. Rizzo, M. Torchiano, N. Mihindukulasooriyac, O. Corcho, R. Garcìa-Castro, "*Completeness and Consistency Analysis for Evolving Knowledge Bases*", 2018.

Appendix A - graphs and data

TOPBRAID	Triples				
Observation n°	**1K**	**10K**	**100K**	**1M**	**5M**
1	2,327	3,475	5,747	20,622	66,816
2	1,995	3,668	5,662	19,836	69,108
3	1,836	3,152	5,719	19,264	68,735
4	2,063	3,562	5,832	20,881	67,528
5	2,254	3,459	6,015	19,529	68,297
6	1,947	3,548	5,861	20,347	69,393
7	2,131	3,503	5,783	22,337	69,471
8	2,175	3,661	5,947	23,542	66,562
9	1,878	3,521	5,661	18,731	68,461
10	2,409	3,374	5,732	20,106	67,349
AVERAGE	2,1015	3,4923	5,7959	20,5195	68,172

Figure 1: validation times for different size datasets required by *TopBraid* out of 10 measurements. Reference unit is seconds.

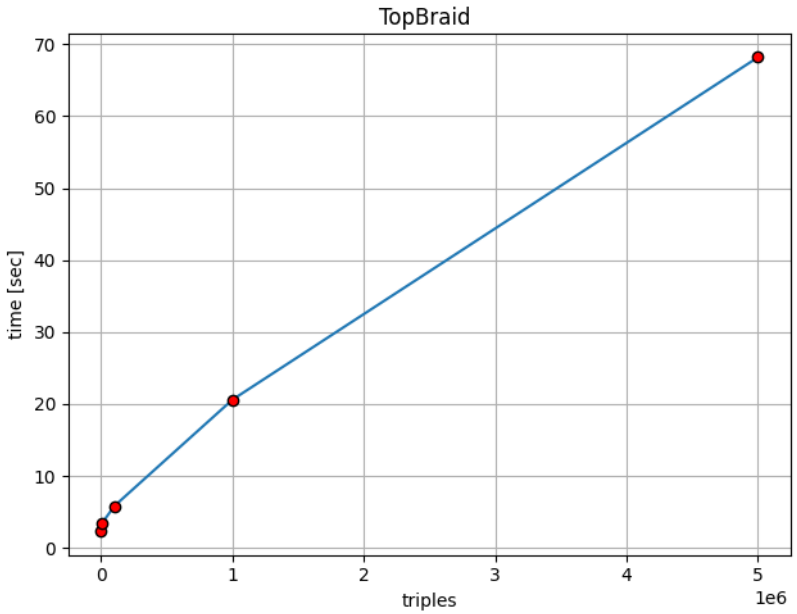

Figure 2: graph of the average validation time required by *TopBraid*.

RDF4J

	Triples				
Observation n°	**1K**	**10K**	**100K**	**1M**	**5M**
1	1,136	1,607	3,285	14,235	64,285
2	1,352	1,564	3,361	13,182	63,581
3	1,194	1,538	3,357	11,534	61,579
4	1,277	1,804	3,284	10,816	62,492
5	1,338	1,447	3,167	13,775	65,068
6	1,098	1,598	3,495	13,397	62,823
7	1,381	1,657	3,367	15,194	63,334
8	1,259	1,633	3,283	14,682	64,772
9	1,164	1,721	3,149	12,683	61,825
10	1,225	1,552	3,315	12,491	63,983
AVERAGE	1,2424	1,6121	3,3063	13,1989	63,3742

Figure 3: validation times for different size datasets required by *RDF4J* out of 10 measurements. Reference unit is seconds.

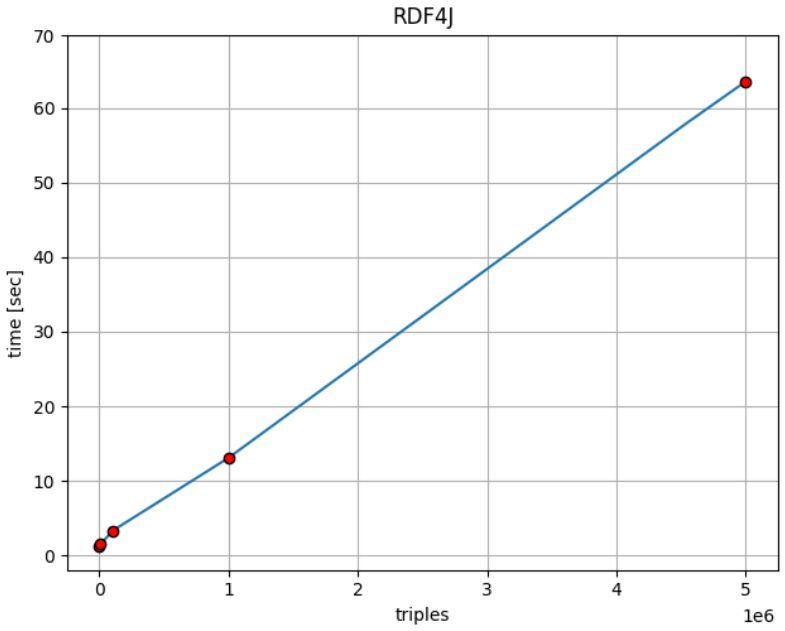

Figure 4: graph of the average validation time required by *RDF4J*.

NEOSEMANTICS					
	Triples				
Observation n°	**1K**	**10K**	**100K**	**1M**	**5M**
1	0,819	2,468	5,396	10,754	31,739
2	1,142	2,372	5,443	10,342	30,309
3	1,038	2,634	5,275	12,391	28,784
4	0,771	2,582	5,581	12,728	28,593
5	0,825	2,491	5,463	11,649	30,861
6	1,159	2,664	5,325	13,102	30,573
7	0,864	2,583	5,294	11,293	29,416
8	1,037	2,358	5,587	12,589	32,119
9	0,745	2,629	5,488	11,755	29,972
10	1,053	2,541	5,491	10,681	31,235
AVERAGE	0,9453	2,5322	5,4343	11,7284	30,3601

Figure 5: validation times for different size datasets required by *Neosemantics* out of 10 measurements. Reference unit is seconds.

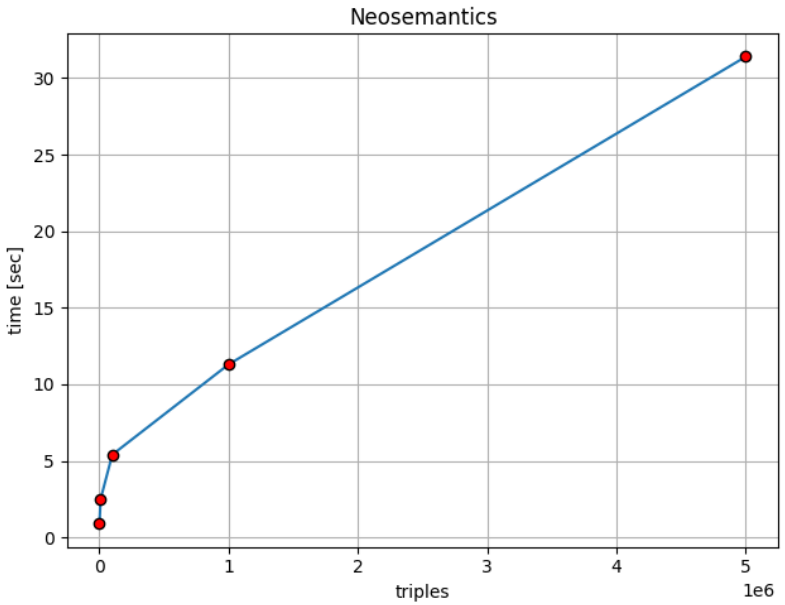

Figure 6: graph of the average validation time required by *Neosemantics*.

AVERAGE VALIDATION TIMES

Triples	TOPBRAID	RDF4J	NEOSEMATICS
1K	2,1015	1,2424	0,9453
10K	3,4923	1,6121	2,5322
100K	5,7959	3,3063	5,4343
1M	20,5195	13,1989	11,7284
5M	68,172	63,3742	30,3601

Figure 7: summary of the average validation times required by the tools.

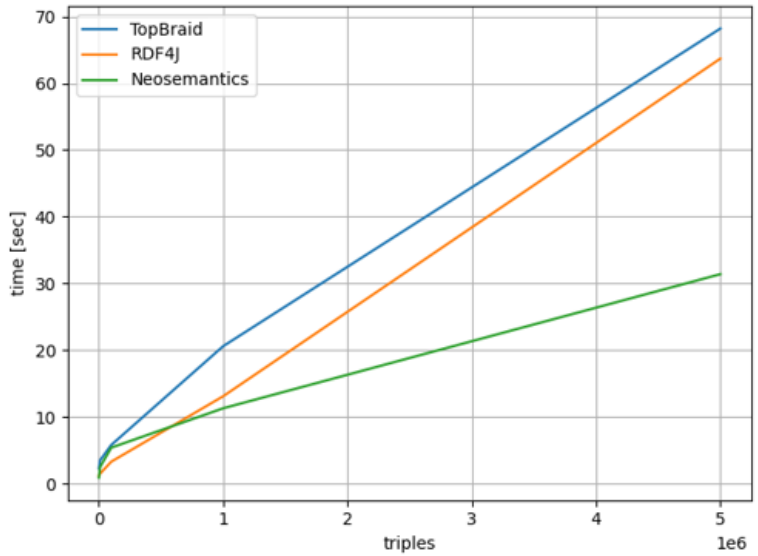

Figure 8: comparison graph of the average validation times required by the tools.

1M triples

Observation n°	TOPBRAID	RDF4J	NEOSEMANTICS
1	1338,7	2631,5	851,8
2	1328,5	2558,2	778,2
3	1206,3	2369,4	976,7
4	1394,1	2357,3	1014,2
5	1242,4	2603,5	910,4
6	1372,6	2584,9	1085,3
7	1451,3	2681,3	896,5
8	1453,8	2655,6	1003,1
9	1186,1	2537,6	943,6
10	1368,9	2485,1	824,3
AVERAGE	1334,27	2546,44	928,41

Figure 9: memory required by the tools during the measurements for 1 million triples. Reference unit is MB of RAM.

5M triples

Observation n°	TOPBRAID	RDF4J	NEOSEMANTICS
1	1986,3	5386,1	1537,4
2	2216,5	5314,7	1417,2
3	2184,2	5184,9	1291,3
4	2096,7	5255,6	1258,4
5	2161,3	5413,4	1493,5
6	2234,8	5289,2	1448,6
7	2291,7	5302,6	1322,1
8	1958,6	5401,2	1587,9
9	2157,3	5236,1	1394,5
10	2052,4	5357,9	1512,9
AVERAGE	2133,98	5314,17	1426,38

Figure 10: memory required by the tools during the measurements for 5 million triples. Reference unit is MB of RAM.

AVERAGE MEMORY USED

Triples	TOPBRAID	RDF4J	NEOSEMATICS
1M	1334,27	2546,44	928,41
5M	2133,98	5314,17	1426,38

Figure 11: summary of the average memory required by the tools for 1M and 5M triples.

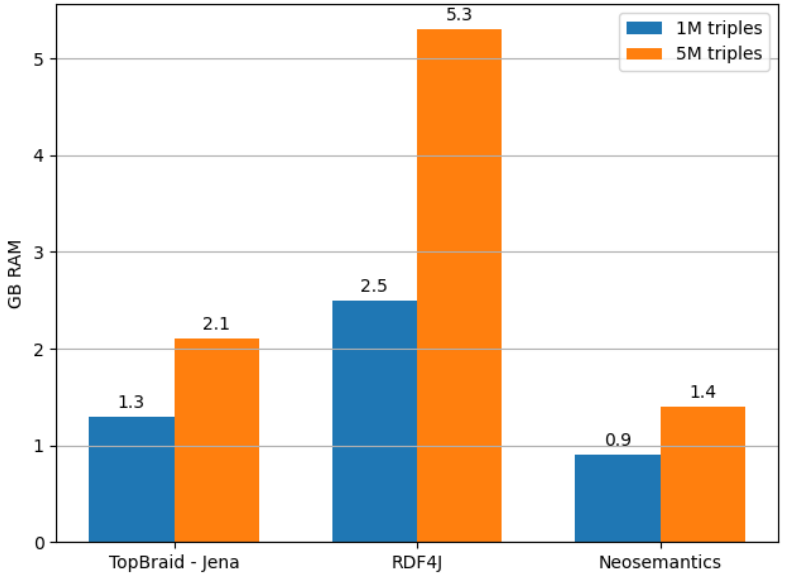

Figure 12: average memory used by the tools for 1M and 5M triples.

Appendix B - violations

- sh:datatype

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/Esselunga> ;
    sh:resultMessage                 "Organisation number of employees has invalid data type" ;
    sh:resultPath                    <http://dbpedia.org/property/numEmployees> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:DatatypeConstraintComponent ;
    sh:sourceShape                   []  ;
    sh:value                         "23.094"^^<http://www.w3.org/2001/XMLSchema#double>
] ;
```

Figure 1: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:node1fe97asp7x942 sh:result _:fefb7230-168c-4123-82f2-1b5026a34f97.

_:fefb7230-168c-4123-82f2-1b5026a34f97 a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Esselunga>;
  sh:value 2.3094E1;
  sh:resultPath <http://dbpedia.org/property/numEmployees>;
  sh:sourceConstraintComponent sh:DatatypeConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fe97at73x43 .

_:node1fe97at73x43 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/numEmployees>;
  sh:datatype <http://www.w3.org/2001/XMLSchema#integer> .
```

Figure 2: violation generated by *RDF4J*.

```
[
  {
    "focusNode": "http://dbpedia.org/resource/Esselunga",
    "nodeType": "http://dbpedia.org/ontology/Organisation",
    "shapeId": "bnode://id/node1fe6rogu8x43",
    "propertyShape": "http://www.w3.org/ns/shacl#DatatypeConstraintComponent",
    "offendingValue": 23.094,
    "resultPath": "http://dbpedia.org/property/numEmployees",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": "property value should be of type http://www.w3.org/2001/XMLSchema#integer"
  }
]
```


Figure 2: violation generated by *Neosematincs* - JSON file.

focusNode	nodeType	shapeId	propertyShape	...
http://dbpedia.org/resource/Esselunga	http://dbpedia.org/ontology/Organisation	bnode://id/node1fb6qj3r9x43	http://www.w3.org/ns/shacl#DatatypeConstraintComponent	

...	offendingValue	resultPath	severity	resultMessage
	23.094	http://dbpedia.org/property/numEmployees	http://www.w3.org/ns/shacl#Violation	property value should be of type http://www.w3.org/2001/XMLSchema#integer

Figure 3: violation generated by *Neosemantics* - table view.

- sh:nodeKind

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/Ordinary_Happiness> ;
    sh:resultMessage                 "Film title is not an IRI or is invalid" ;
    sh:resultPath                    <http://dbpedia.org/property/title> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:NodeKindConstraintComponent ;
    sh:sourceShape                   [] ;
    sh:value                         "Ordinary Happiness"@en
] ;
```

Figure 4: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:417f4970-9795-43f2-a609-fffedab3f51a a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Ordinary_Happiness>;
  sh:value "Ordinary Happiness"@en;
  sh:resultPath <http://dbpedia.org/property/title>;
  sh:sourceConstraintComponent sh:NodeKindConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fh2pr4nix1 .

_:node1fh2pr4nix1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/title>;
  sh:nodeKind sh:IRI .
```

Figure 5: violation generated by *RDF4J*.

```
[
 {
    "focusNode": "http://dbpedia.org/resource/Ordinary_Happiness",
    "nodeType": "http://dbpedia.org/ontology/Film",
    "shapeId": "bnode://id/node1fe6rogu8x31",
    "propertyShape": "http://www.w3.org/ns/shacl#NodeKindConstraintComponent",
    "offendingValue": "Ordinary Happiness",
    "resultPath": "http://dbpedia.org/property/title",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": ""
  }
]
```


Figure 6: violation generated by *Neosemantics* - JSON file.

focusNode	nodeType	shapeId	propertyShape	...
http://dbpedia.org/resource/Ordinary_Happiness	http://dbpedia.org/ontology/Film	bnode://id/node1fb6qj3r9x31	http://www.w3.org/ns/shacl#NodeKindConstraintComponent	

...

offendingValue	resultPath	severity	resultMessage
Ordinary Happiness	http://dbpedia.org/property/title	http://www.w3.org/ns/shacl#Violation	

Figure 7: violation generated by *Neosemantics* - table view.

- sh:minCount - sh:maxCount

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                   sh:ValidationResult ;
    sh:focusNode                        <http://dbpedia.org/resource/Chōfu_Airport> ;
    sh:resultMessage                    "Airport IATA code is invalid or missing" ;
    sh:resultPath                       <http://dbpedia.org/property/iata> ;
    sh:resultSeverity                   sh:Violation ;
    sh:sourceConstraintComponent        sh:MinCountConstraintComponent ;
    sh:sourceShape                      [] ;
] ;
```

Figure 8: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:3636add0-70ac-4f7b-9585-3a4d10bd87dd a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Chōfu_Airport>;
  sh:resultPath <http://dbpedia.org/property/iata>;
  sh:sourceConstraintComponent sh:MinCountConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fe6quvgvx52 .

_:node1fe6quvgvx52 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/iata>;
  sh:minCount 1 .
```

Figure 9: violation generated by *RDF4J*.

```
[
  {
    "focusNode": "http://dbpedia.org/resource/Chōfu_Airport",
    "nodeType": "http://dbpedia.org/ontology/Airport",
    "shapeId": "bnode://id/node1fe6rogu8x52",
    "propertyShape": "http://www.w3.org/ns/shacl#MinCountConstraintComponent",
    "offendingValue": null,
    "resultPath": "http://dbpedia.org/property/iata",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": "unnacceptable cardinality: 0"
  }
]
```


Figure 10: violation generated by *Neosemantics* - JSON file.

focusNode	nodeType	shapeId	propertyShape	...
http://dbpedia.org/resource/Chōfu_Airport	http://dbpedia.org/ontology/Airport	bnode://id/node1fe6rogu8x52	http://www.w3.org/ns/shacl#MinCountConstraintComponent	

...	offendingValue	resultPath	severity	resultMessage
		http://dbpedia.org/property/iata	http://www.w3.org/ns/shacl#Violation	unnacceptable cardinality: 0

Figure 11: violation generated by *Neosemantics* - table view.

- sh:minInclusive - sh:maxInclusive and sh:minExclusive - sh:maxExclusive

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                   sh:ValidationResult ;
    sh:focusNode                        <http://dbpedia.org/resource/Gorgan_Airport> ;
    sh:resultMessage                    "Place latitude not in standard range" ;
    sh:resultPath                       <http://www.w3.org/2003/01/geo/wgs84_pos#lat> ;
    sh:resultSeverity                   sh:Violation ;
    sh:sourceConstraintComponent        sh:MinInclusiveConstraintComponent ;
    sh:sourceShape                      [] ;
    sh:value                            "-100.0000"^^<http://www.w3.org/2001/XMLSchema#float>
] ;
```

Figure 12: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:32d19fe8-8efc-4c8d-bfab-34bc9653a26e a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Gorgan_Airport>;
  sh:value "-100.0000"^^<http://www.w3.org/2001/XMLSchema#float>;
  sh:resultPath <http://www.w3.org/2003/01/geo/wgs84_pos#lat>;
  sh:sourceConstraintComponent sh:MinInclusiveConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ffn5bqv1x1 .

_:node1ffn5bqv1x1 a sh:PropertyShape;
  sh:path <http://www.w3.org/2003/01/geo/wgs84_pos#lat>;
  sh:minInclusive -90.0 .
```

Figure 13: violation generated by *RDF4J*.

```
[
    {
    "focusNode": "http://dbpedia.org/resource/Gorgan_Airport",
    "nodeType": "http://dbpedia.org/ontology/Airport",
    "shapeId": "bnode://id/node1ffn468vfx1",
    "propertyShape": "http://www.w3.org/ns/shacl#MinExclusiveConstraintComponent",
    "offendingValue": -100.0000,
    "resultPath": "http://www.w3.org/2003/01/geo/wgs84_pos#lat",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": ""
  }
]
```


Figure 14: violation generated by *Neosemantics* - JSON file.

focusNode	nodeType	shapeId	propertyShape	...
http://dbpedia.org/resource/Gorgan_Airport	http://dbpedia.org/ontology/Airport	bnode://id/node1ffn468vfx1	http://www.w3.org/ns/shacl#MinExclusiveConstraintComponent	

...

offendingValue	resultPath	severity	resultMessage
-100	http://www.w3.org/2003/01/geo/wgs84_pos#lat	http://www.w3.org/ns/shacl#Violation	

Figure 15: violation generated by *Neosemantics* - table view.

- sh:minLength - sh:maxLength

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/Gorgan_Airport> ;
    sh:resultMessage                 "Airport ICAO code is invalid or missing" ;
    sh:resultPath                    <http://dbpedia.org/property/icao> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:MaxLengthConstraintComponent ;
    sh:sourceShape                   [] ;
    sh:value                         "OINGX"@en
] ;
```

Figure 16: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:67e25c77-267c-470b-8ff1-24ce76a3d049 a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Gorgan_Airport>;
  sh:value "OINGX"@en;
  sh:resultPath <http://dbpedia.org/property/icao>;
  sh:sourceConstraintComponent sh:MaxLengthConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ffn1esnfx2 .

_:node1ffn1esnfx2 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/icao>;
  sh:maxLength 4 .
```

Figure 17: violation generated by *RDF4J*.

```
[
    {
    "focusNode": "http://dbpedia.org/resource/Gorgan_Airport",
    "nodeType": "http://dbpedia.org/ontology/Airport",
    "shapeId": "bnode://id/node1ffn1p6a7x2",
    "propertyShape": "http://www.w3.org/ns/shacl#MaxLengthConstraintComponent",
    "offendingValue": "OINGX",
    "resultPath": "http://dbpedia.org/property/icao",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": ""
  }
]
```


Figure 18: violation generated by *Neosemantics* - JSON file.

focusNode	nodeType	shapeId	...
http://dbpedia.org/resource/Gorgan_Airport	http://dbpedia.org/ontology/Airport	bnode://id/node1ffn1p6a7x2	

...	propertyShape	offendingValue	resultPath	severity	resultMessage
	http://www.w3.org/ns/shacl#MaxLengthConstraintComponent	OINGX	http://dbpedia.org/property/icao	http://www.w3.org/ns/shacl#Violation	

Figure 19: violation generated by *Neosemantics* - table view.

- sh:pattern

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/José_Enrique_Varela> ;
    sh:resultMessage                 "Person birth date has invalid format" ;
    sh:resultPath                    <http://dbpedia.org/property/birthDate> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:PatternConstraintComponent ;
    sh:sourceShape                   [] ;
    sh:value                         "1891-1-1"^^<http://www.w3.org/2001/XMLSchema#date>
] ;
```

Figure 20: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:c07e8e34-86a1-4bf9-bf96-288470727abe a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/José_Enrique_Varela>;
  sh:value "1891-1-1"^^<http://www.w3.org/2001/XMLSchema#date>;
  sh:resultPath <http://dbpedia.org/property/birthDate>;
  sh:sourceConstraintComponent sh:PatternConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fh3c0kgnx1 .

_:node1fh3c0kgnx1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/birthDate>;
  sh:pattern "^\\d{4}-\\d{2}-\\d{2}$" .
```

Figure 21: violation generated by *RDF4J*.

```
[
 {
    "focusNode": "http://dbpedia.org/resource/José_Enrique_Varela",
    "nodeType": "http://dbpedia.org/ontology/Person",
    "shapeId": "bnode://id/node1fh3bhs39x1",
    "propertyShape": "http://www.w3.org/ns/shacl#PatternConstraintComponent",
    "offendingValue": "1891-1-1",
    "resultPath": "http://dbpedia.org/property/birthDate",
    "severity": "http://www.w3.org/ns/shacl#Violation",
    "resultMessage": ""
  }
]
```


Figure 22: violation generated by *Neosemantics* - JSON file.

focusNode	nodeType	shapeId	...
http://dbpedia.org/resource/José_Enrique_Varela	http://dbpedia.org/ontology/Person	bnode://id/node1fh3bhs39x1	

...	propertyShape	offendingValue	resultPath	severity	resultMessage
	http://www.w3.org/ns/shacl#PatternConstraintComponent	1891-1-1	http://dbpedia.org/property/birthDate	http://www.w3.org/ns/shacl#Violation	

Figure 23: violation generated by *Neosemantics* - table view.

- sh:languageIn

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                   sh:ValidationResult ;
    sh:focusNode                        <http://dbpedia.org/resource/Clash_by_Night> ;
    sh:resultMessage                    "Film title is undefined or invalid" ;
    sh:resultPath                       <http://dbpedia.org/property/title>;
    sh:resultSeverity                   sh:Violation ;
    sh:sourceConstraintComponent        sh:LanguageInConstraintComponent ;
    sh:sourceShape                      [] ;
    sh:value                            "Clash by Night"@en;
] ;
```

Figure 24: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:1ac39aca-1b2c-4a4b-aae1-73e4c357bb06 a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Clash_by_Night>;
  sh:value "Clash by Night"@en;
  sh:resultPath <http://dbpedia.org/property/title>;
  sh:sourceConstraintComponent sh:LanguageInConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ffn2gcomx1 .

_:node1ffn2gcomx1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/title>;
  sh:languageIn _:node1ffn2gcomx2 .

_:node1ffn2gcomx2 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "it";
```

Figure 25: violation generated by *RDF4J*.

- sh:equals

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     ex:bob ;
    sh:resultMessage                 "Must have same values as ex:firstName" ;
    sh:resultPath                    ex:givenName ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:EqualsConstraintComponent ;
    sh:sourceShape                   _:b0 ;
    sh:value                         "Robert"
] ;
```

Figure 26: violation generated by *TopBraid*.

- sh:disjoint

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                               sh:ValidationResult ;
    sh:focusNode                    ex:carol ;
    sh:resultMessage                "Property must not share any values with ex:lastName" ;
    sh:resultPath                   ex:givenName ;
    sh:resultSeverity               sh:Violation ;
    sh:sourceConstraintComponent    sh:DisjointConstraintComponent ;
    sh:sourceShape                  [] ;
    sh:value                        "Carol"
] ;
```

Figure 27: violation generated by *TopBraid*.

- sh:lessThan

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/Walter_Schuck> ;
    sh:resultMessage                 "Person birth date is greater than death date" ;
    sh:resultPath                    <http://dbpedia.org/property/birthDate> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:LessThanConstraintComponent ;
    sh:sourceShape                   [] ;
    sh:value                         "2000-07-30"^^<http://www.w3.org/2001/XMLSchema#date>
] ;
```

Figure 28: violation generated by *TopBraid*.

- sh:or

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/The_Fatal_Woman> ;
    sh:resultMessage                 "Value has none of the shapes from the 'or' list" ;
    sh:resultPath                    <http://dbpedia.org/property/released> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:OrConstraintComponent ;
    sh:sourceShape                   []  ;
    sh:value                         "1.9"^^<http://www.w3.org/2001/XMLSchema#double>
]
```

Figure 29: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:75848eef-eeb6-4f26-8411-c4ea826d41b6 a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/The_Fatal_Woman>;
  sh:value 1.9E0;
  sh:resultPath <http://dbpedia.org/property/released>;
  sh:sourceConstraintComponent sh:OrConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ff2uefglx1 .

_:node1ff2uefglx1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/released>;
  sh:or _:node1ff2uefglx2 .

_:node1ff2uefglx3 a sh:NodeShape;
  sh:datatype <http://www.w3.org/2001/XMLSchema#date>;
  sh:pattern "^\\d{4}-\\d{2}-\\d{2}$" .

_:node1ff2uefglx5 a sh:NodeShape;
  sh:datatype <http://www.w3.org/2001/XMLSchema#integer>;
  sh:pattern "^\\d{4}" .
```

Figure 30: violation generated by *RDF4J*.

- sh:and

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     <http://dbpedia.org/resource/Ordinary_Happiness> ;
    sh:resultMessage                 "Film title is invalid or undefined" ;
    sh:resultPath                    <http://dbpedia.org/property/title> ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:AndConstraintComponent ;
    sh:sourceShape                   [] ;
    sh:value                         "Ordinary Happiness"@en
] ;
```

Figure 31: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:32b009cd-a2e9-4126-8dfe-df0e0a55b76e a sh:ValidationResult;
  sh:focusNode <http://dbpedia.org/resource/Ordinary_Happiness>;
  sh:value "Ordinary Happiness"@en;
  sh:resultPath <http://dbpedia.org/property/title>;
  sh:sourceConstraintComponent sh:AndConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1ffkumfsmx1 .

_:node1ffkumfsmx1 a sh:PropertyShape;
  sh:path <http://dbpedia.org/property/title>;
  sh:and _:node1ffkumfsmx2 .

_:node1ffkumfsmx3 a sh:NodeShape;
  sh:nodeKind sh:Literal .

_:node1ffkumfsmx5 a sh:NodeShape;
  sh:languageIn _:node1ffkumfsmx6 .

_:node1ffkumfsmx6 <http://www.w3.org/1999/02/22-rdf-syntax-ns#first> "it";
```

Figure 32: violation generated by *RDF4J*.

- sh:qualifiedValueShape and sh:qualifiedMinCount - sh:qualifiedMaxCount

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                                sh:ValidationResult ;
    sh:focusNode                     ex:hand ;
    sh:resultMessage                 "Less than 1 values, not well-formed thumb" ;
    sh:resultPath                    ex:digit ;
    sh:resultSeverity                sh:Violation ;
    sh:sourceConstraintComponent     sh:QualifiedMinCountConstraintComponent ;
    sh:sourceShape                   [] ;
] ;
```

Figura 33: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:c102361d-4e87-4b78-9092-8e276b1199bc a sh:ValidationResult;
  sh:focusNode <http://example.org/ns#emily>;
  sh:resultPath <http://schema.org/parent>;
  sh:sourceConstraintComponent sh:QualifiedMinCountConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fh3fff0ox1 .

_:node1fh3fff0ox1 a sh:PropertyShape;
  sh:path <http://schema.org/parent>;
  sh:qualifiedValueShape _:node1fh3fff0ox2;
  sh:qualifiedMinCount 1 .
```

Figure 34: violation generetaed by *RDF4J*.

- sh:qualifiedValueShapesDisjoint

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ex: <http://SEKM_EXAM.com/ns#> .

sh:result [
    a                               sh:ValidationResult ;
    sh:focusNode                    ex:hand ;
    sh:resultMessage                "Less than 1 values, not well-formed thumb" ;
    sh:resultPath                   ex:digit ;
    sh:resultSeverity               sh:Violation ;
    sh:sourceConstraintComponent    sh:QualifiedMinCountConstraintComponent ;
    sh:sourceShape                  [] ;
] ;
```

Figure 35: violation generated by *TopBraid*.

```
@prefix sh: <http://www.w3.org/ns/shacl#> .

_:84917a1e-b197-4100-b17a-ef0068babb5b a sh:ValidationResult;
  sh:focusNode <http://example.org/ns#hand2>;
  sh:resultPath <http://example.org/ns#digit>;
  sh:sourceConstraintComponent sh:QualifiedMaxCountConstraintComponent;
  sh:resultSeverity sh:Violation;
  sh:sourceShape _:node1fh3ie8uqx3 .

_:node1fh3ie8uqx3 a sh:PropertyShape;
  sh:path <http://example.org/ns#digit>;
  sh:qualifiedValueShape _:node1fh3ie8uqx4;
  sh:qualifiedValueShapesDisjoint true;
  sh:qualifiedMaxCount 4 .

_:node1fh3ie8uqx4 a sh:NodeShape;
  sh:class <http://example.org/ns#Finger> .
```

Figure 36: violation generated by *RDF4J*.