Anti-Patterns for explaining inconsistencies in large knowledge graphs

Thomas de Groot[1], Joe Raad[1,2], Stefan Schlobach[1]

[1] Vrije Universiteit Amsterdam [2] Université Paris-Saclay

IC 2022

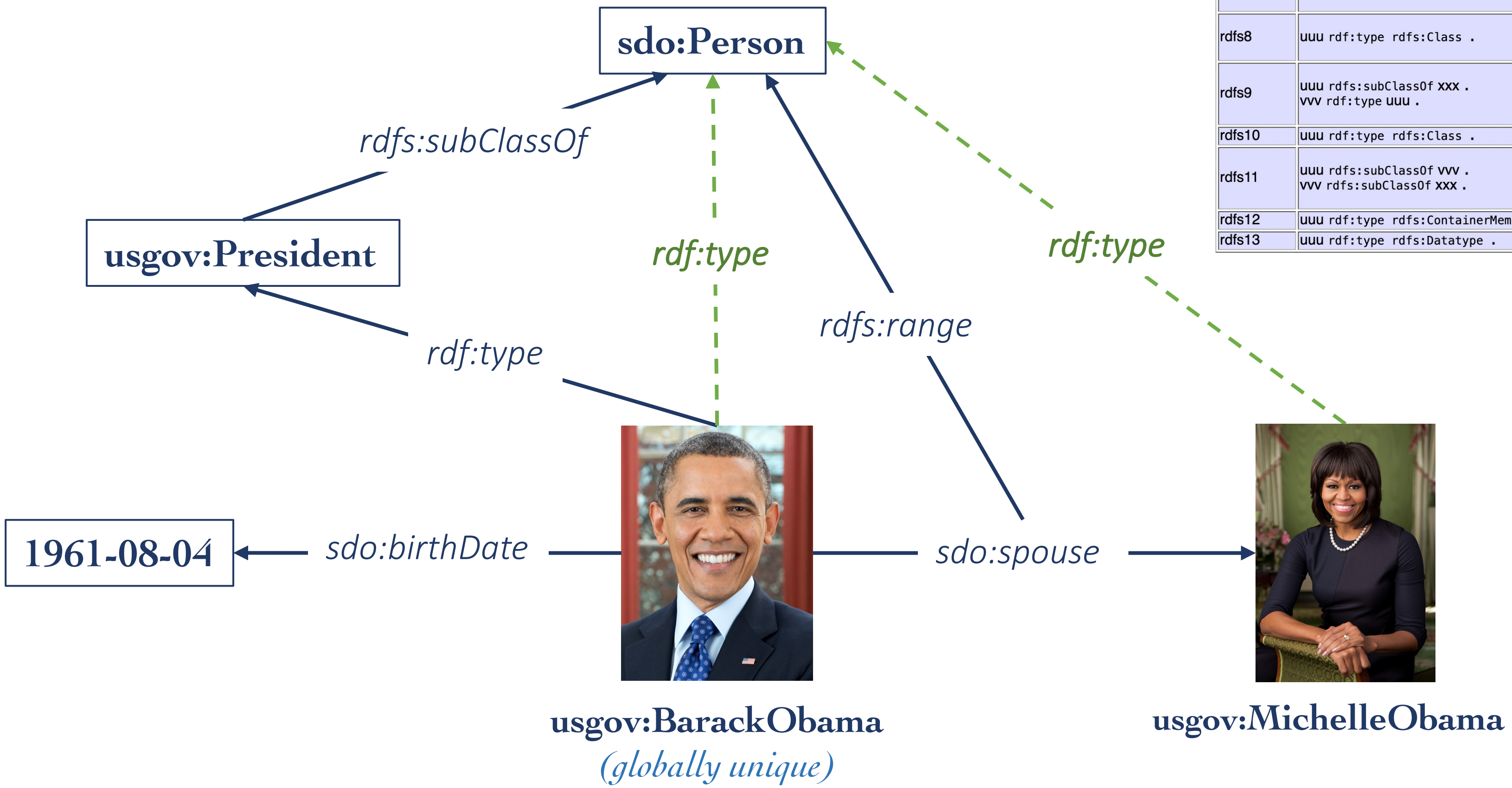
(published at ESWC 2021)

Knowledge Graphs (KG)

7.3 RDFS Entailment Rules

RDFS entailment rules.

Rule Name	If E contains:	then add:
rdfs1	uuu aaa lll. where lll is a plain literal (with or without a language tag).	_:nnn rdf:type rdfs:Literal . where _:nnn identifies a blank node allocated to lll by rule rule lg.
rdfs2	aaa rdfs:domain xxx . uuu aaa yyy .	uuu rdf:type xxx .
rdfs3	aaa rdfs:range xxx . uuu aaa vvv .	vvv rdf:type xxx .
rdfs4a	uuu aaa xxx .	uuu rdf:type rdfs:Resource .
rdfs4b	uuu aaa vvv.	vvv rdf:type rdfs:Resource .
rdfs5	uuu rdfs:subPropertyOf vvv . vvv rdfs:subPropertyOf xxx .	uuu rdfs:subPropertyOf xxx .
rdfs6	uuu rdf:type rdf:Property .	uuu rdfs:subPropertyOf uuu .
rdfs7	aaa rdfs:subPropertyOf bbb . uuu aaa yyy .	uuu bbb yyy .
rdfs8	uuu rdf:type rdfs:Class .	uuu rdfs:subClassOf rdfs:Resource .
rdfs9	uuu rdfs:subClassOf xxx . vvv rdf:type uuu .	vvv rdf:type xxx .
rdfs10	uuu rdf:type rdfs:Class .	uuu rdfs:subClassOf uuu .
rdfs11	uuu rdfs:subClassOf vvv . vvv rdfs:subClassOf xxx .	uuu rdfs:subClassOf xxx .
rdfs12	uuu rdf:type rdfs:ContainerMembershipProperty .	uuu rdfs:subPropertyOf rdfs:member .
rdfs13	uuu rdf:type rdfs:Datatype .	uuu rdfs:subClassOf rdfs:Literal .



Logical Inconsistencies

- Example from the Pizza Ontology (that serves as tutorial for OWL and Protégé)
- Contains two contradictions asserted by its developers on purpose

Explanation 1 ☐ Display laconic explanation

Explanation for: CheesyVegetableTopping EquivalentTo owl:Nothing

CheesyVegetableTopping **SubClassOf** CheeseTopping

CheesyVegetableTopping **SubClassOf** VegetableTopping

DisjointClasses: CheeseTopping, SeafoodTopping, FruitTopping, HerbSpiceTopping, MeatTopping, NutTopping, SauceTopping, VegetableTopping

Explanation 1 ☐ Display laconic explanation

Explanation for: IceCream EquivalentTo owl:Nothing

IceCream **SubClassOf** hasTopping **some** FruitTopping

hasTopping **Domain** Pizza

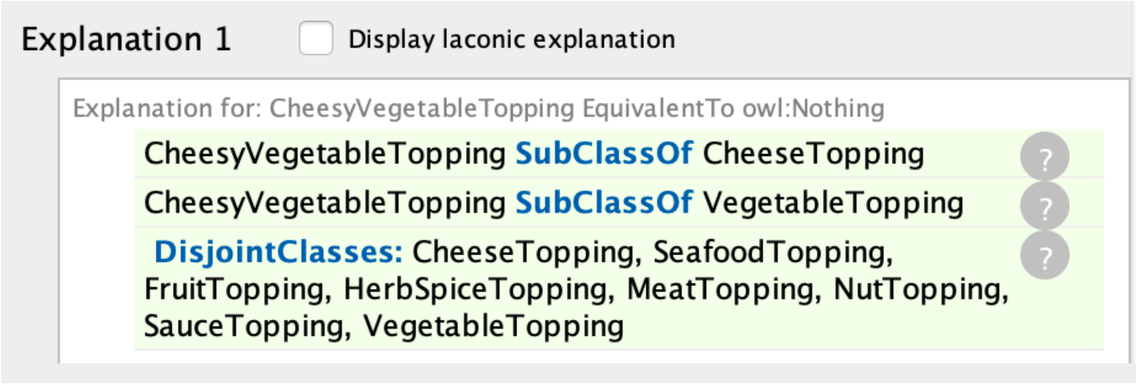
DisjointClasses: IceCream, Pizza, PizzaBase, PizzaTopping

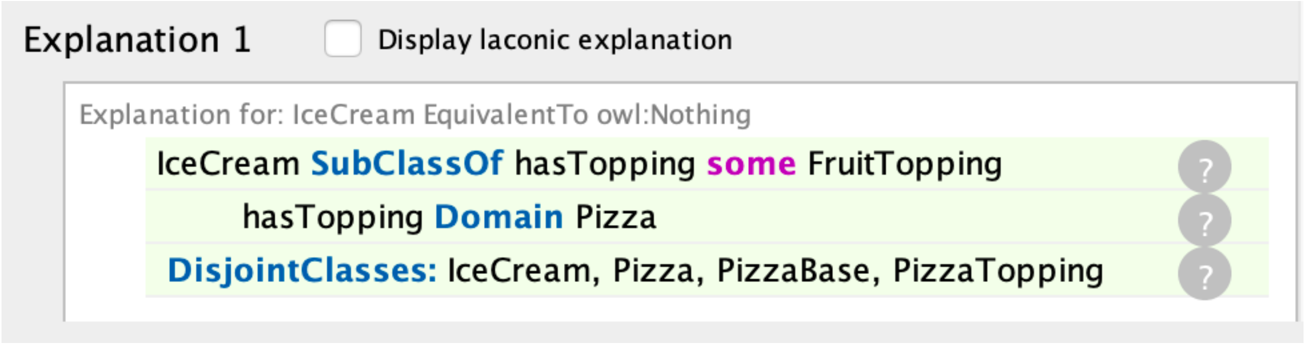
When these unsatisfiable classes are instantiated → Inconsistency

Inconsistent KG : a graph for which no model exists

Logical Inconsistencies

- Example from the Pizza Ontology (that serves as tutorial for OWL and Protégé)
- Contains two contradictions asserted by its developers on purpose





Justifications

Minimal subset of the KG that is sufficient for the entailment (contradiction) to hold

Explaining inconsistencies in large KGs

Limits of justifications :

- Can occur frequently
- Domain dependent
- Justification retrieval algorithms do not scale to billions of triples

Motivation for this work

Develop an approach that allows us to :

- Detect and explain inconsistencies in large KGs
- Analyse and compare different KGs' qualities
- Facilitate repairing the inconsistencies

Contributions

1. Definition of an anti-pattern : a more general explanation for contradictions that categorizes common mistakes in KGs, independently from their domain

2. Approach that allows to retrieve these generalized explanations from any KG, independently from their size

3. Comparison of the consistency of the most commonly used KGs in the Web

1. Anti-Patterns

Anti-pattern

formal definition

Analysing Large Inconsistent Knowledge Graphs using Anti-Patterns

Thomas de Groot, Joe Raad, and Stefan Schlobach

Department of Computer Science, Vrije Universiteit Amsterdam, The Netherlands
{t.j.a.de.groot | j.raad | k.s.schlobach}@vu.nl

Abstract. A number of Knowledge Graphs (KGs) on the Web of Data contain contradicting statements, and therefore are logically inconsistent. This makes reasoning limited and the knowledge formally useless. Understanding how these contradictions are formed, how often they occur, and how they vary between different KGs is essential for fixing such contradictions, or developing better tools that handle inconsistent KGs. Methods exist to explain a single contradiction, by finding the minimal set of axioms sufficient to produce it, a process known as justification retrieval. In large KGs, these justifications can be frequent and might redundantly refer to the same type of modelling mistake. Furthermore, these justifications are –by definition– domain dependent, and hence difficult to interpret or compare. This paper uses the notion of anti-pattern for generalising these justifications, and presents an approach for detecting almost all anti-patterns from any inconsistent KG. Experiments on KGs of over 28 billion triples show the scalability of this approach, and the benefits of anti-patterns for analysing and comparing logical errors between different KGs.

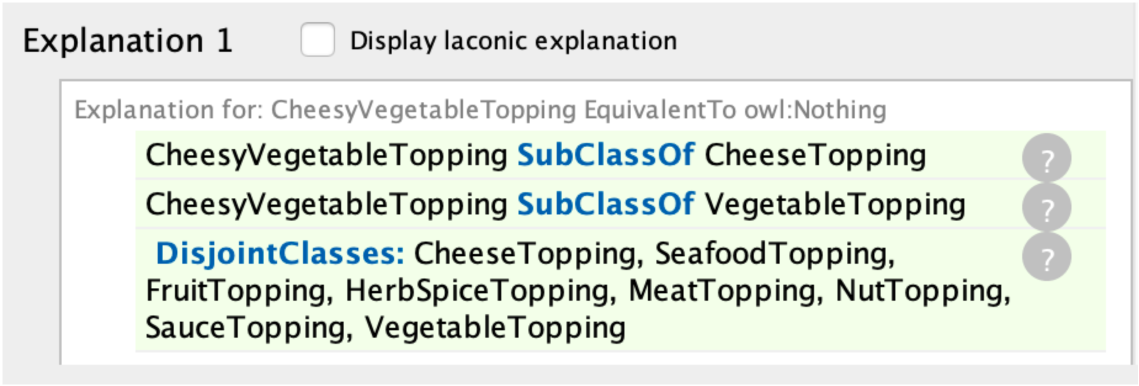
Keywords: linked open data, reasoning, inconsistency

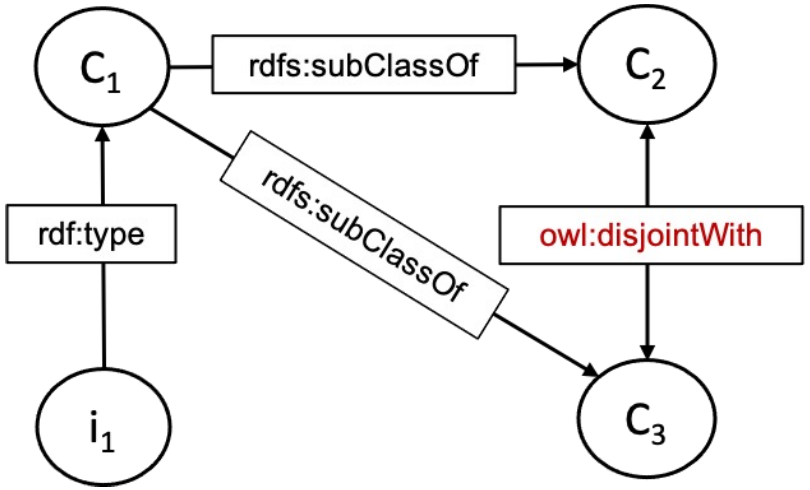
1 Introduction

Through the combination of web technologies and a judicious choice of formal expressivity (description logics which are based on decidable 2-variable fragments of first order logic), it has become possible to construct and reason over Knowledge Graphs (KGs) of sizes that were not imaginable only few years ago. Nowadays, KGs of billions of statements are routinely deployed by researchers from various fields and companies. Since most of the large KGs are traditionally built over a longer period of time, by different collaborators, these KGs are highly prone for containing logically contradicting statements. As a consequence, reasoning over these KGs becomes limited and the knowledge formally useless.

Typically, once these contradicting statements in a KG are retrieved, they are either logically explained [22] and repaired [20], or ignored via non-standard reasoning [13]. This work falls in the first category of approaches where the focus is to find and explain what has been stated in the KG that causes the inconsistency. Understanding how these contradictions are formed and how often

- Basic Graph Pattern (contains variables, similarly to a SPARQL query)
- Generalization of common type of inconsistency in a KG
- Replaces domain-dependent information in a justification with variables





Anti-pattern

formal definition

Analysing Large Inconsistent Knowledge Graphs using Anti-Patterns

Thomas de Groot, Joe Raad, and Stefan Schlobach

Department of Computer Science, Vrije Universiteit Amsterdam, The Netherlands
{t.j.a.de.groot | j.raad | k.s.schlobach}@vu.nl

Abstract. A number of Knowledge Graphs (KGs) on the Web of Data contain contradicting statements, and therefore are logically inconsistent. This makes reasoning limited and the knowledge formally useless. Understanding how these contradictions are formed, how often they occur, and how they vary between different KGs is essential for fixing such contradictions, or developing better tools that handle inconsistent KGs. Methods exist to explain a single contradiction, by finding the minimal set of axioms sufficient to produce it, a process known as justification retrieval. In large KGs, these justifications can be frequent and might redundantly refer to the same type of modelling mistake. Furthermore, these justifications are –by definition– domain dependent, and hence difficult to interpret or compare. This paper uses the notion of anti-pattern for generalising these justifications, and presents an approach for detecting almost all anti-patterns from any inconsistent KG. Experiments on KGs of over 28 billion triples show the scalability of this approach, and the benefits of anti-patterns for analysing and comparing logical errors between different KGs.

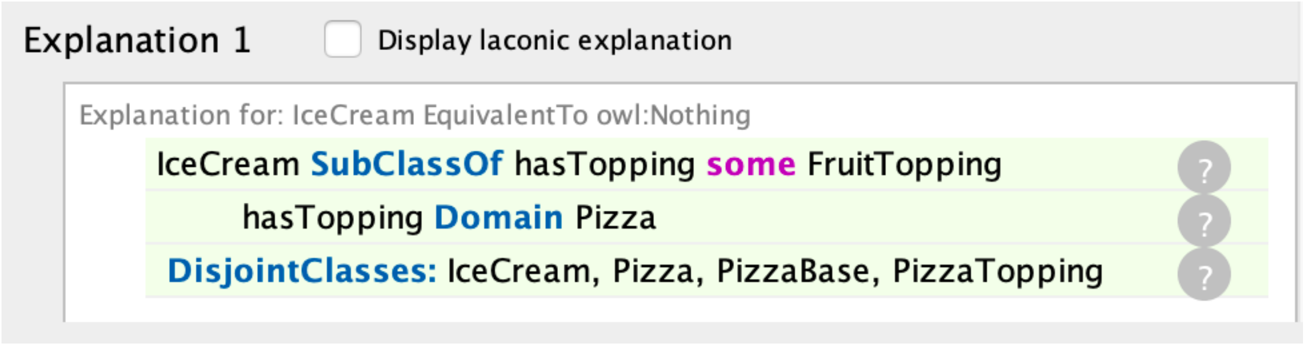
Keywords: linked open data, reasoning, inconsistency

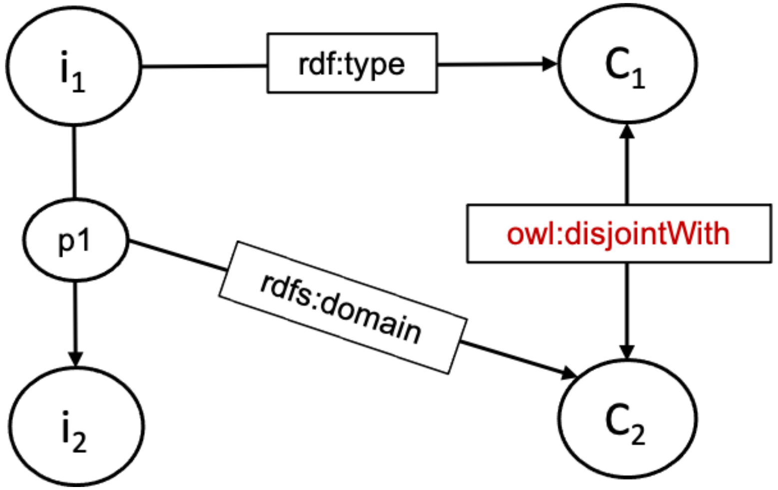
1 Introduction

Through the combination of web technologies and a judicious choice of formal expressivity (description logics which are based on decidable 2-variable fragments of first order logic), it has become possible to construct and reason over Knowledge Graphs (KGs) of sizes that were not imaginable only few years ago. Nowadays, KGs of billions of statements are routinely deployed by researchers from various fields and companies. Since most of the large KGs are traditionally built over a longer period of time, by different collaborators, these KGs are highly prone for containing logically contradicting statements. As a consequence, reasoning over these KGs becomes limited and the knowledge formally useless.

Typically, once these contradicting statements in a KG are retrieved, they are either logically explained [22] and repaired [20], or ignored via non-standard reasoning [13]. This work falls in the first category of approaches where the focus is to find and explain what has been stated in the KG that causes the inconsistency. Understanding how these contradictions are formed and how often

- Basic Graph Pattern (contains variables, similarly to a SPARQL query)
- Generalization of common type of inconsistency in a KG
- Replaces domain-dependent information in a justification with variables





2. Detection of Anti-Patterns in large KGs

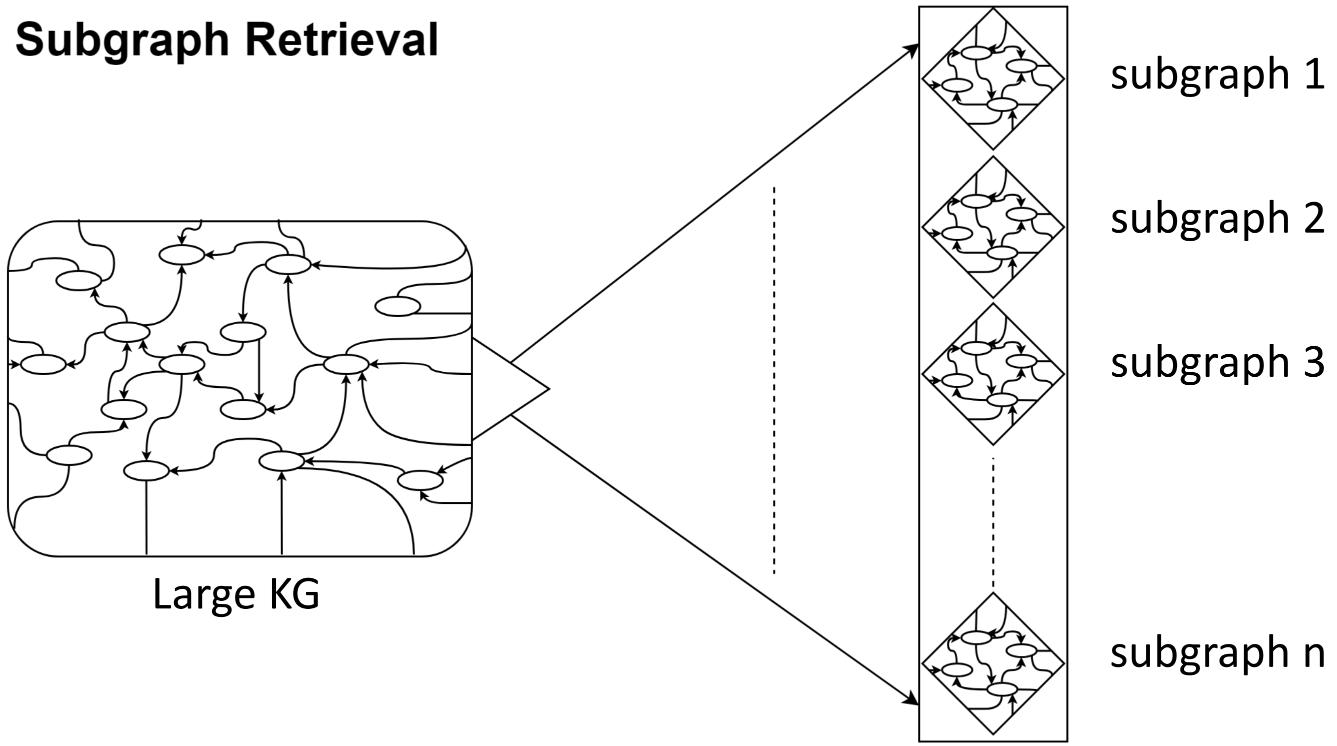
Anti-pattern detection

1. Retrieve justifications of contradictions
2. Generalise these detected justifications into anti-patterns

Complexity :

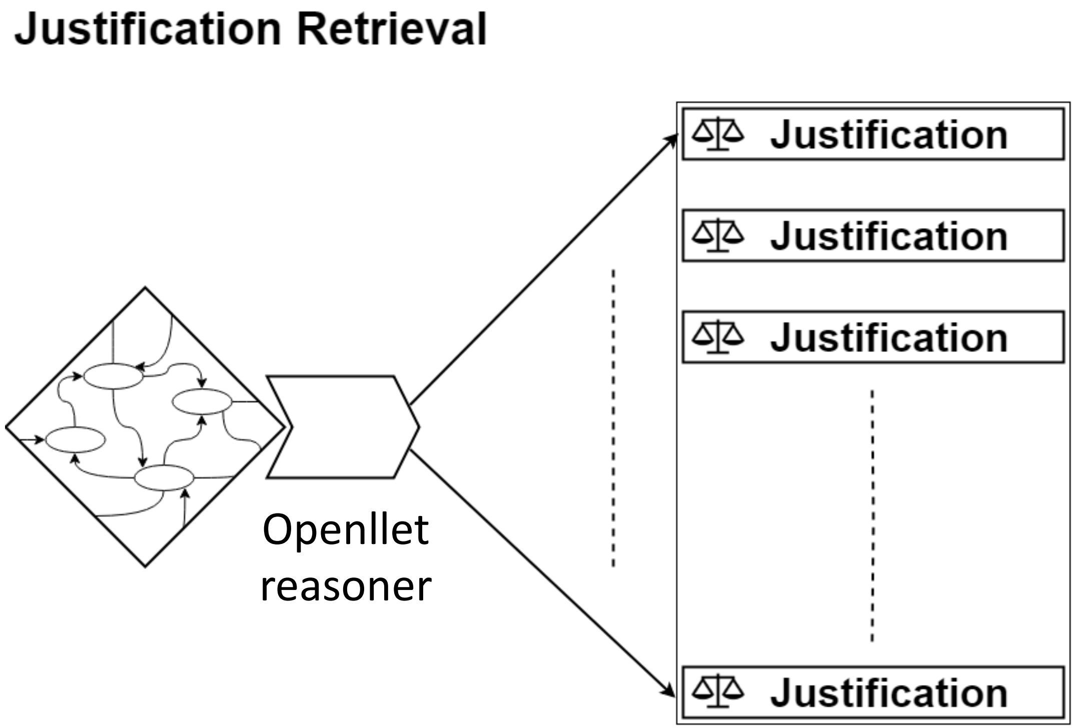
- KGs can be too large to query or store in memory
- Justification retrieval algorithms do not scale
- Guaranteeing the retrieval of all anti-patterns in a large KG

Step 1



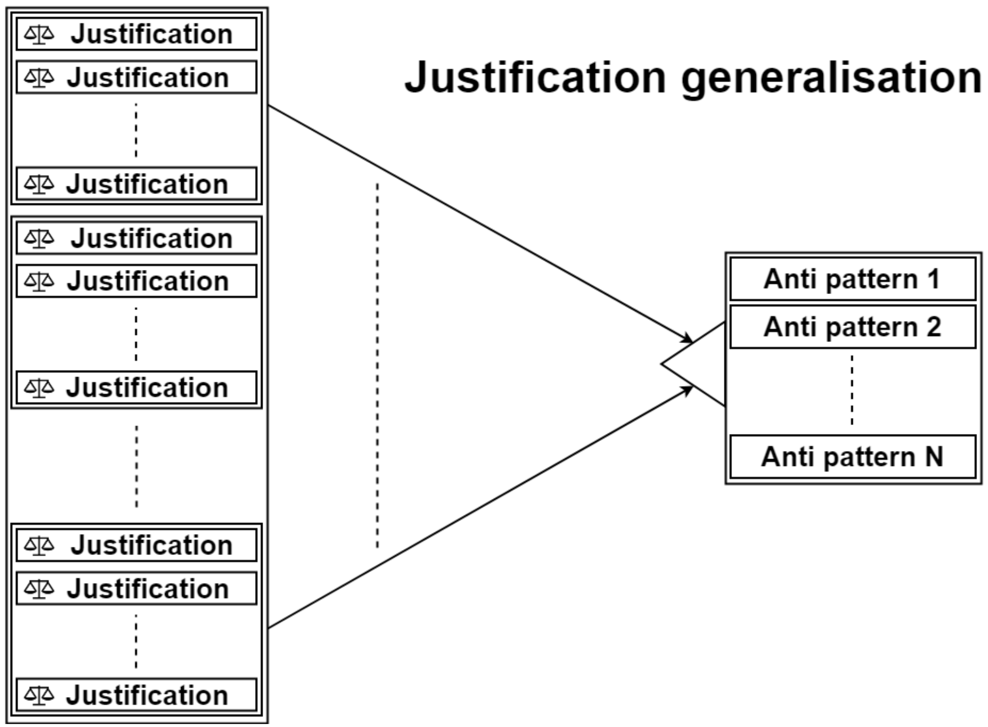
Retrieval of subgraphs from the knowledge graph with a max limit of G_{max} triples

Step 2



Retrieval of the detected inconsistencies with their justifications in each subgraph

Step 3

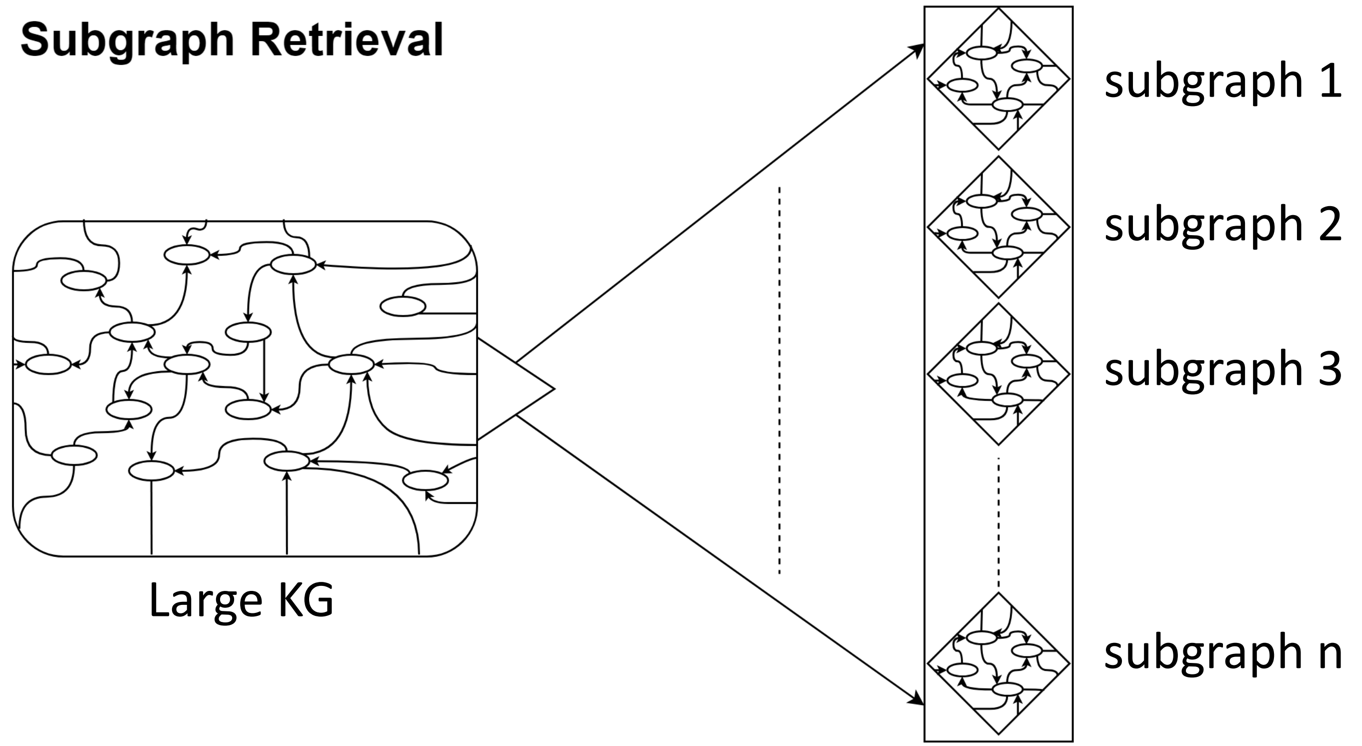


Generalization of each justification into an anti-pattern
(and grouping similar anti-patterns using graph isomorphism)

3. Experiments

i. Completeness Evaluation

Completeness Evaluation



Retrieval of subgraphs from the knowledge graph with a max limit of G_{max} triples

Pizza ontology

Table 1: Impact of the subgraph size limit G_{max} on the number of detected anti-patterns and the runtime of the approach (in seconds) for the Pizza dataset.

G_{max}	Detected Anti-patterns	Total Runtime	Step 1 Runtime	Step 2 Runtime	Step 3 Runtime	Number of Subgraphs
50	2	3	1	2	0.01	335
100	2	4.3	1.3	3	0.01	186
250	2	8	3	5	0.05	77
500	2	13	6	7	0.04	38
750	2	18	8	10	0.08	25
1K	2	23	10	13	0.08	19
No limit	2	3.2	-	3.1	0.1	-

Result 1 : Both contradictions (and anti-patterns) can be detected despite graph partition

Linked Open Vocabulary + YAGO

Table 2: Impact of the subgraph size limit G_{max} on the number of detected anti-patterns and the runtime of the approach (in seconds) for LOV and YAGO.

	G_{max}	Detected Anti-patterns	Total Runtime	Step 1 Runtime	Step 2 Runtime	Step 3 Runtime	Number of Subgraphs
LOV	*500*	0	1,783	216	1,566	2	101,673
	1K	2	3,505	429	3,073	3	50,960
	5K	39	4,525	668	3,829	28	10,218
	10K	39	5,106	739	4,349	18	5,109
	25K	39	5,347	835	4,493	18	2,041
	50K	39	5,497	858	4,615	24	1,014
	100K	39	5,758	946	4,792	20	507
YAGO	*500*	0	3,403	649	2,753	1	18,203,648
	1K	0	39,41	1,223	2,717	1	9,123,936
	5K	135	14,342	2,125	12,004	214	1,829,442
	10K	135	18,283	2,265	15,739	279	914,721
	25K	135	19,174	2,938	16,013	223	365,422
	50K	135	34,177	3,289	30,684	204	181,547
	100K	135	68,264	3,976	64,081	206	90,773

Result 2 : Partitioning the graph into subgraphs of maximum 5K triples gives the best trade-off between coverage and runtime

3. Experiments

ii. Scalability Evaluation

Scalability Evaluation

Table 3: Results of detecting anti-patterns from three of the largest KGs in the Web: LOD-a-lot, DBpedia and YAGO.

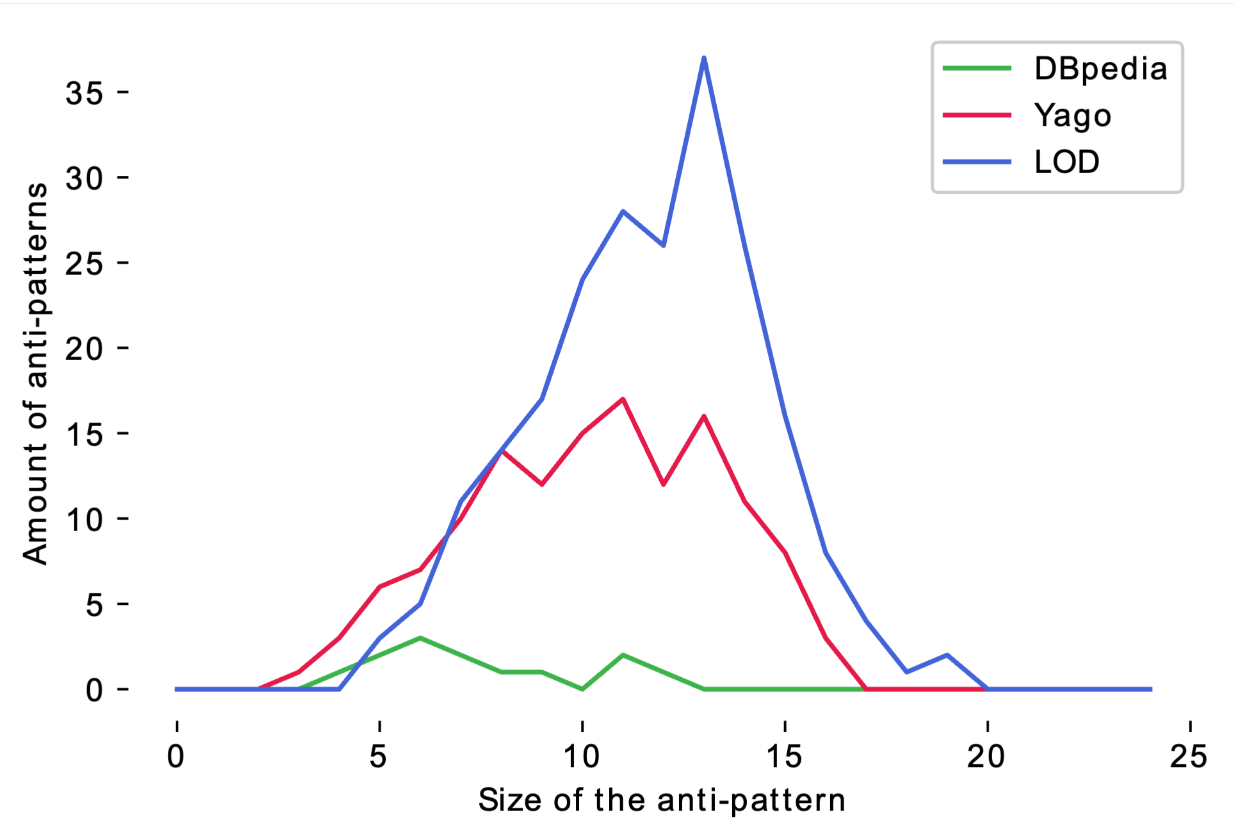
	LOD-a-lot	**DBpedia**	**YAGO**
number of triples	28,362,198,927	1,040,358,853	158,991,568
number of distinct namespaces	9,619	20	11
number of distinct anti-patterns	222	13	135
largest anti-pattern size	19	12	16
runtime (in hours)	*157.56*	*13.01*	*3.98*

Result 3 : We can detect almost all anti-patterns from any KG with their justifications (in a reasonable time, on a standard server)

3. Experiments

iii. Analysis of contradictions on the Web

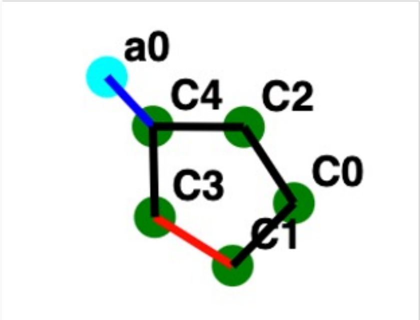
Most common sizes of anti-patterns



Result 4 : Most inconsistencies in DBpedia stem from direct instantiations of unsatisfiable classes, while in LOD-a-lot and YAGO it requires following longer chains

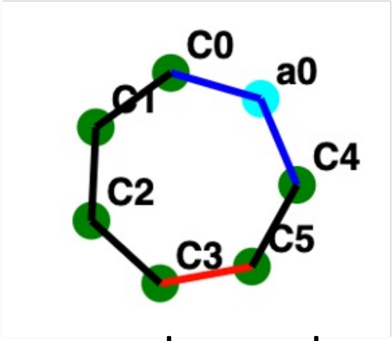
Most common types of anti-patterns

```
SELECT * WHERE {
?C4 rdfs:subClassOf ?C3.
?C4 rdfs:subClassOf ?C2.
?C2 rdfs:subClassOf ?C0.
?C0 rdfs:subClassOf ?C1.
?a0 rdf:type ?C4.
?C1 owl:disjointWith ?C3. }
```



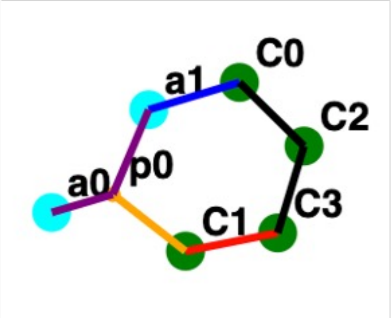
kite graph

```
SELECT * WHERE {
?a0 rdf:type ?C0.
?a0 rdf:type ?C4.
?C3 owl:disjointWith ?C5.
?C2 rdfs:subClassOf ?C3.
?C4 rdfs:subClassOf ?C5.
?C0 rdfs:subClassOf ?C1.
?C1 rdfs:subClassOf ?C2.}
```



cycle graph

```
SELECT * WHERE {
?a1 rdf:type ?C0.
?C4 owl:disjointWith ?C1.
?p0 rdfs:range ?C1.
?C0 rdfs:subClassOf ?C2.
?C2 rdfs:subClassOf ?C3.
?C3 rdfs:subClassOf ?C4.
?a0 ?p0 ?a1.}
```



domain or range based graph

Result 5 : Three common types of anti-patterns (when transitive chains are shortened)

Anti-patterns vs justifications

Table 5: Impact of generalising justifications to anti-patterns

$sup(P)$	**LOD-a-lot**	**DBpedia**	**YAGO**
Minimum	2	1	1
Maximum	45,935,769	32,997	379,546
Average	4,988,176.9	7,796.07	133,998.31
Median	23,126	4,469	106,698
Total	1,107,375,273	101,349	18,089,773
Total per triple	3.9%	0.009%	11.3%

Result 6 : There exists over 1 billion justifications of contradictions in LOD-a-lot

Result 7 : A single anti-pattern in LOD-a-lot generalizes over 45M retrieved justifications

Conclusion

Contributions

1. **Definition of an anti-pattern :** a more general explanation for contradictions that categorizes common mistakes in KGs, independently from their domain

2. **Anti-pattern retrieval approach** that allows to retrieve these generalized explanations from any KG, independently from their size

3. **Analysis of contradictions in the Web** by comparing anti-patterns detected in real-world datasets totaling 30 billion triples

All detected anti-patterns with their support are available online as SPARQL queries
https://tinyurl.com/ic2022-antipatterns

These inconsistencies can now be repaired using a CONSTRUCT query

Contributions

1. **Definition of an anti-pattern :** a more general explanation for contradictions that categorizes common mistakes in KGs, independently from their domain

2. **Anti-pattern retrieval approach** that allows to retrieve these generalized explanations from any KG, independently from their size

3. **Analysis of contradictions in the Web** by comparing anti-patterns detected on real-world datasets totaling 30 billion triples

---

Thank you

Thomas de Groot   Joe Raad   Stefan Schlobach

Analysing Large Inconsistent Knowledge Graphs using Anti-Patterns, ESWC 2021