Can we learn how to "Repair” Knowledge Graphs using constraints?

Axel Polleres, WU Vienna

Starting points/Base materials:

- https://ww101.ai.wu.ac.at/
- Tutorial: **Wikidata Wizardry 101: From Query Spells to Data Charms (ISWC2024)**
  - Includes many Jupyter/GoogleColab notebooks about Wikidata and constraints in Wikidata

- https://github.com/nicolasferranti/wikidata-constraints-formalization
  - Formalization of Wikidata constraints in SPARQL and SHACL
  - Accompanying paper: https://www.semantic-web-journal.net/content/formalizing-and-validating-wikidatas-property-constraints-using-shacl-and-sparql-0

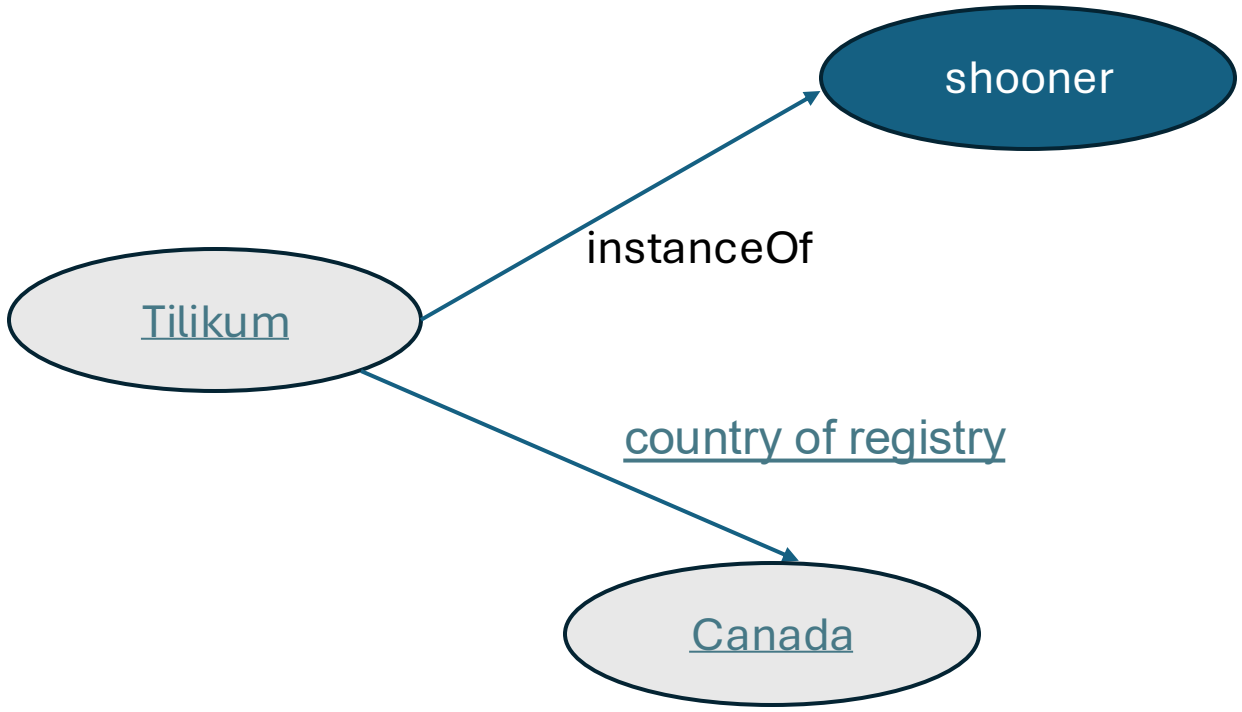
- https://colab.tuwien.ac.at/spaces/PBA/pages/339183111/Lecture+Series
  - My talk “Querying Knowledge Graphs, Graph Learning and LLMs” on 22.05.2025

- Great talk by Miguel 03.07.205 about “Knowledge Graphs and learning”, and various tasks e.g.
  - Link prediction
  - Node classification…

  We will talk about a slightly different task here: Graph repairs wrt Constraints!

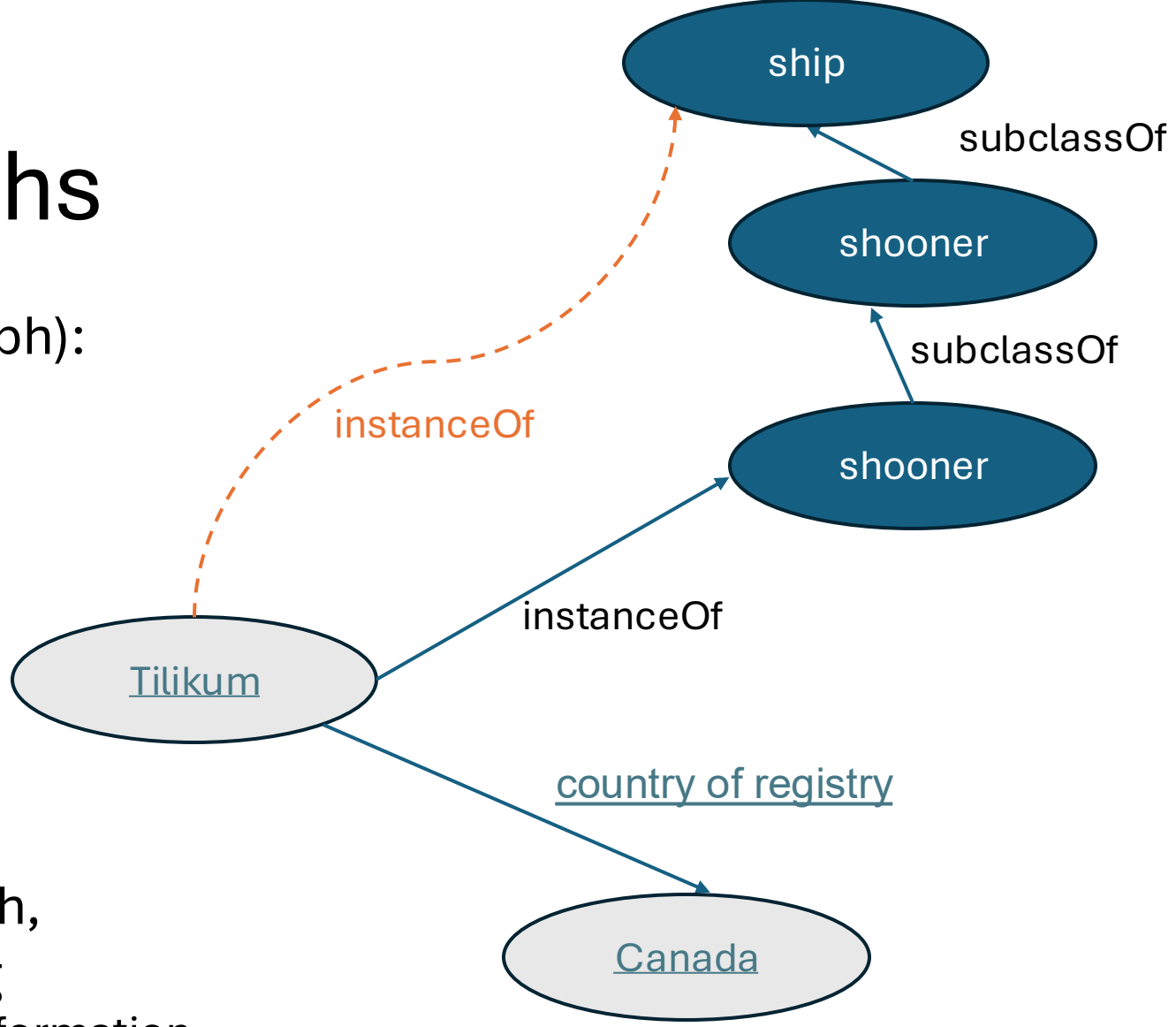
Knowledge Graphs

- Simple (directed, labelled graph):



Knowledge Graphs

- Simple (directed, labelled graph):

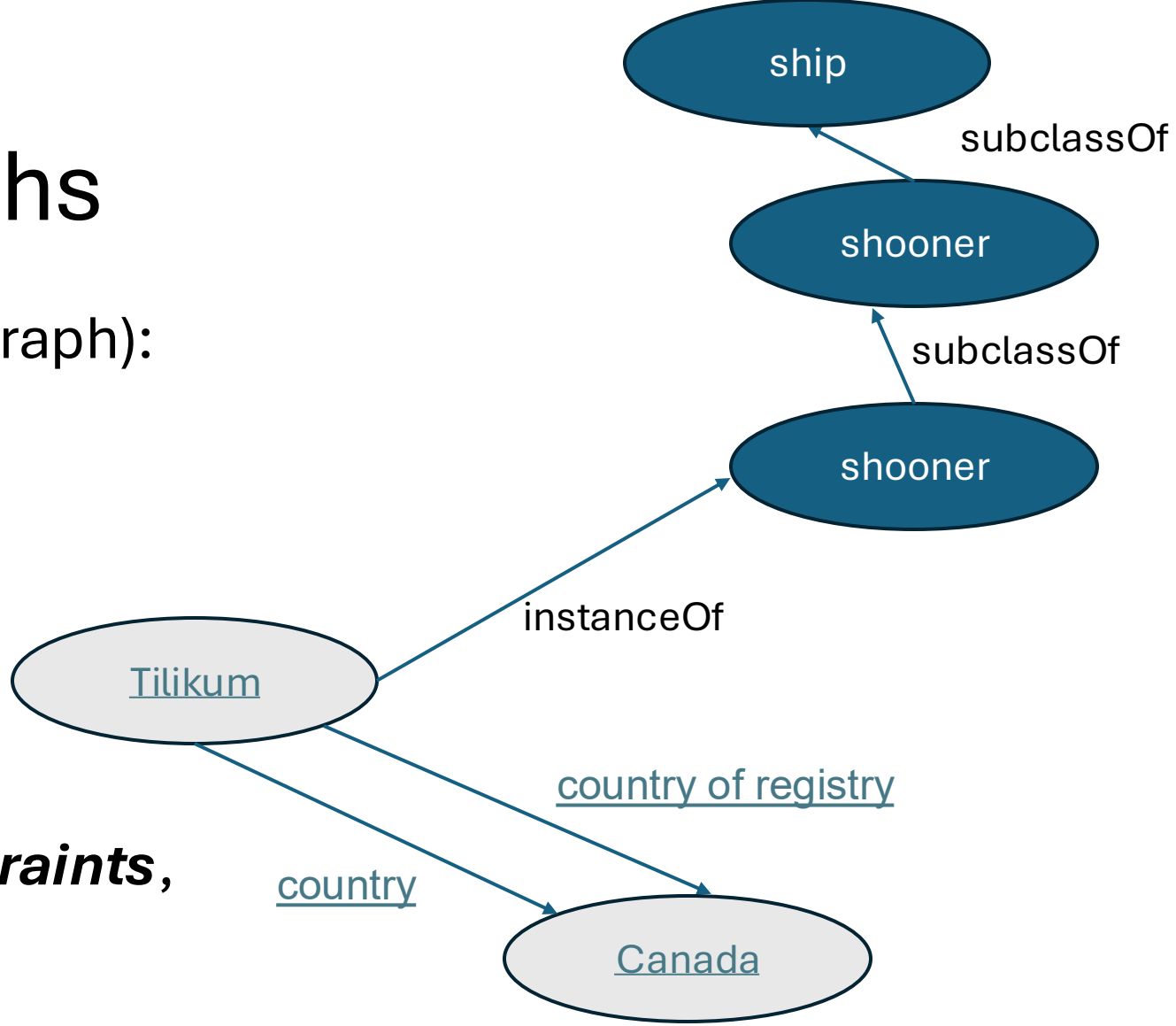


"Ontologogy" is part of the graph,

- typically "deductive" meaning
  - i.e., allows to infer additional information

Knowledge Graphs

- Simple (directed, labelled graph):

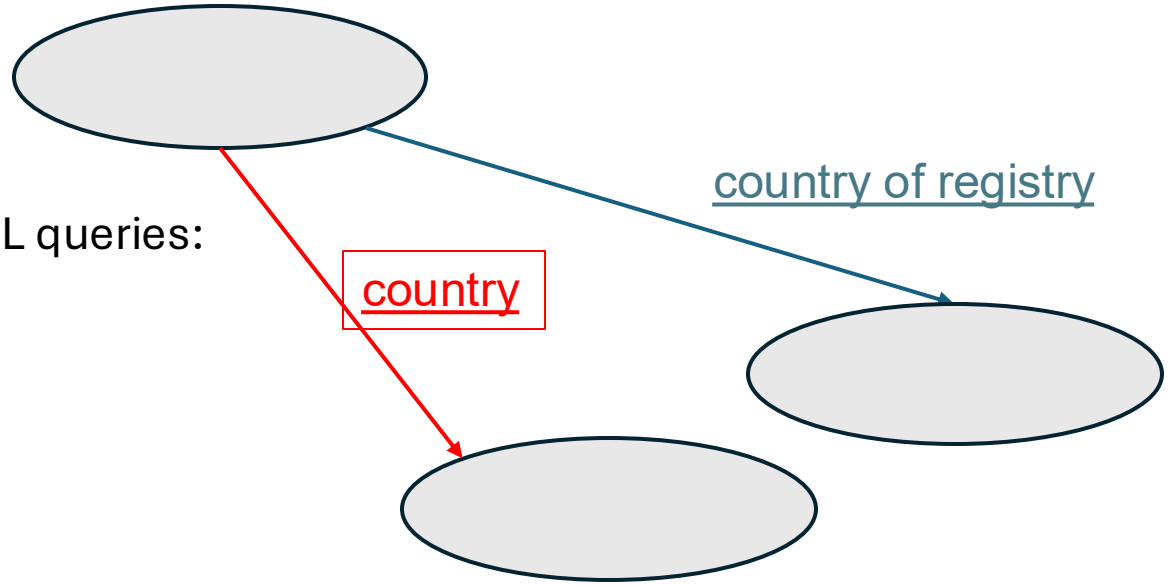


- Could have ***property constraints***, such as:

- *"country conflicts-with country of registry"*
  - *"i.e., an entity should not have both a "country" and a "country of registry"*

Checking constraints

- Check Constraints on Graphs by looking for “graph patterns”



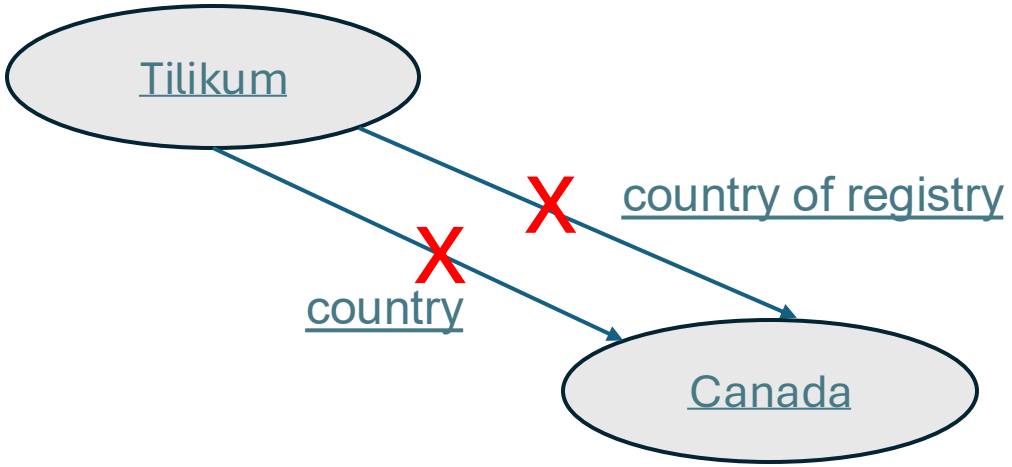
Graph patterns can be checked by SPARQL queries:

```
SELECT * {
?P1 conflicts-with ?P2 .
?S ?P1 ?X.
?S ?P2 ?Y.
}
```

ATTENTION: This is a simplification, cf. https://github.com/nicolasferranti/wikidata-constraints-formalization/

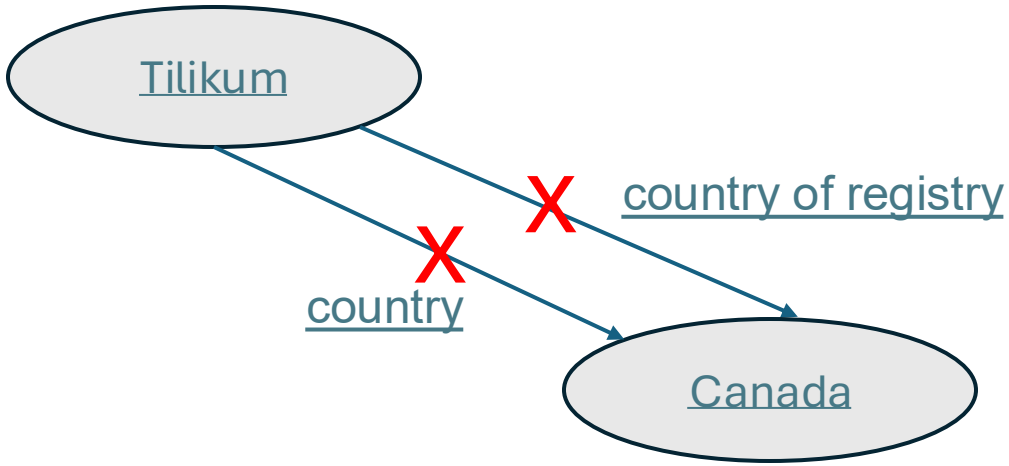
https://github.com/nicolasferranti/wikidata-constraints-formalization/blob/main/constraints-formalization/conflicts-with%20/conflicts-with.rq

Knowledge Graphs



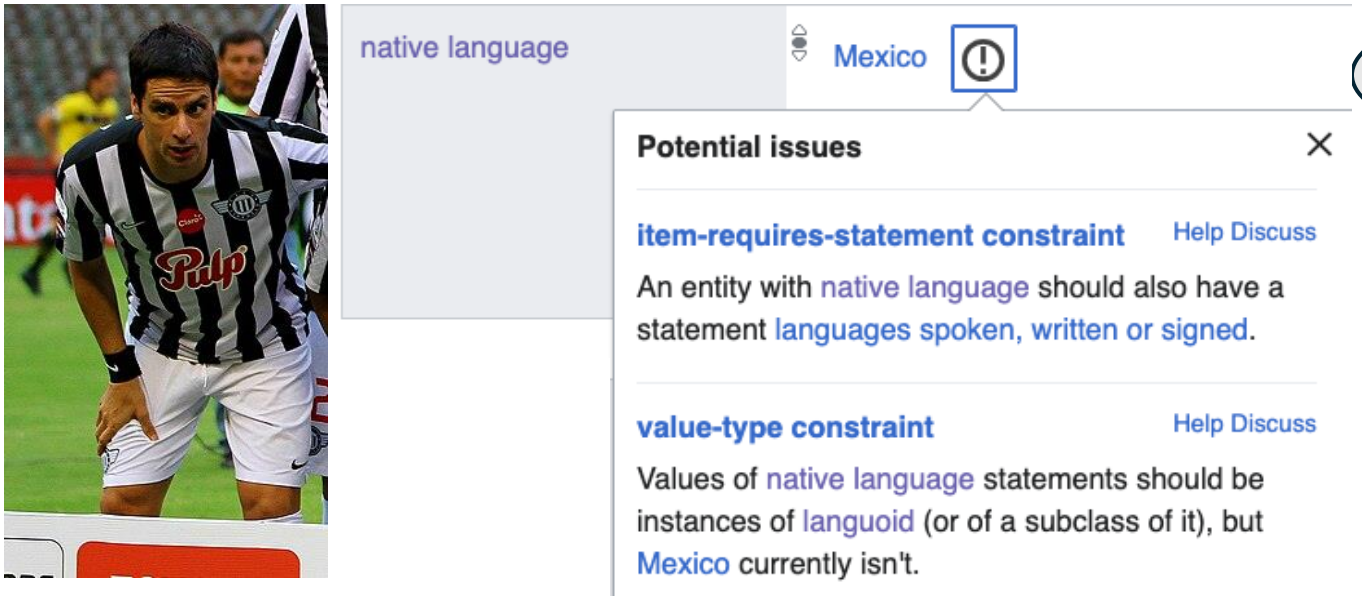
Constraint Repairs

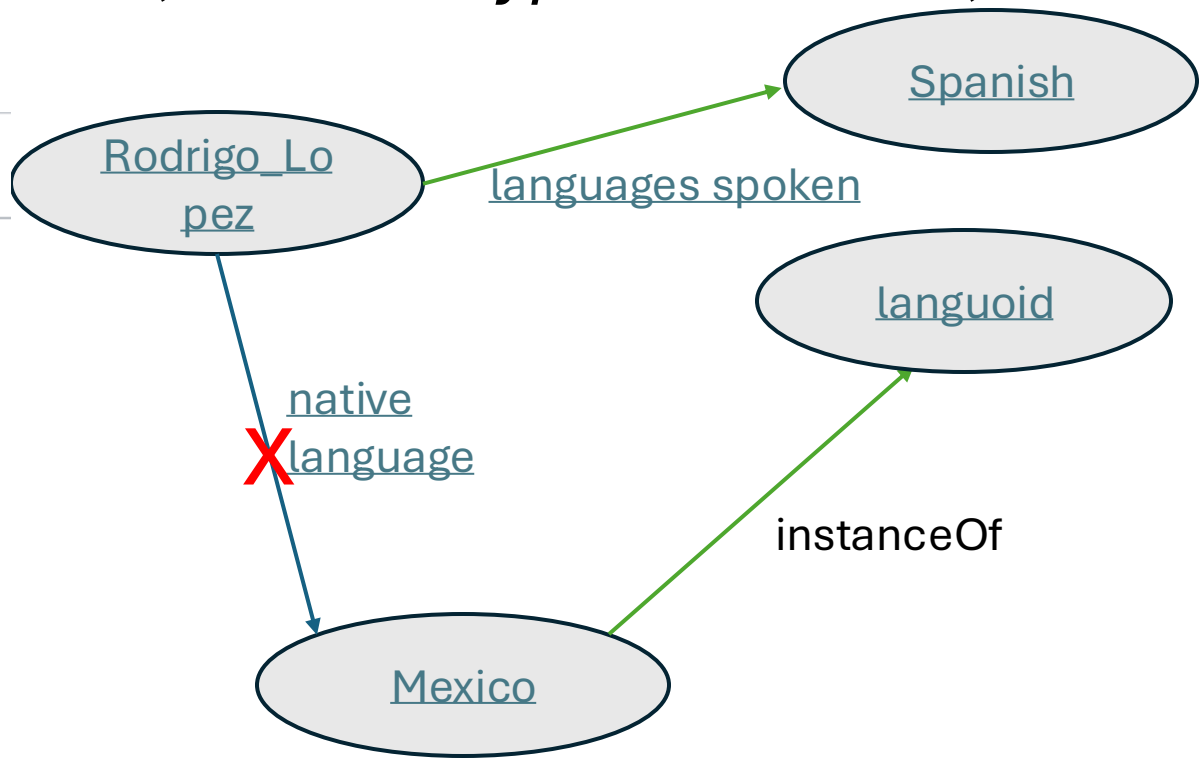
- Additions or deletions of missing information.
- For conflicts-with, e.g. deletion of either:



Constraint Repairs

- Additions or deletions of missing information.
- For *item-required-statement* constraint, or *value-type-constraint*, e.g. deletion or **addition**:





Question: can we learn how to repair a graph by GNNs?

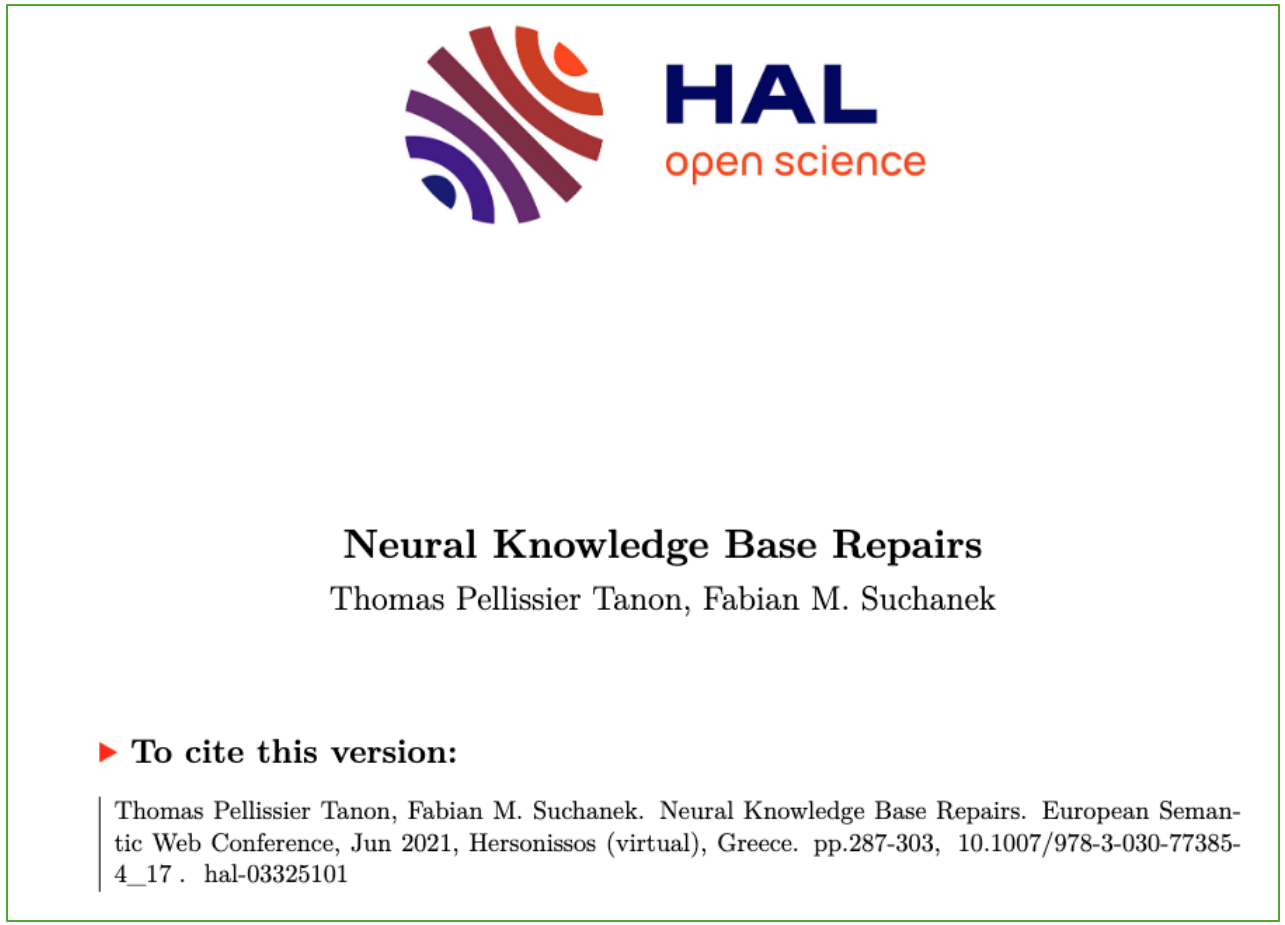
Group Task:

- Understand the training data
- Try to implement a GNN that learns such repairs, e.g. start with something small, e.g.
  - conflicts with,
  - item-requires-statement,
  - type, Value-type,
  - (single-value)

  constraints
- Try to understand what it can (and can’t?) learn

Good thing: We have training data!

https://hal.science/hal-03325101v1/



HAL open science

Neural Knowledge Base Repairs

Thomas Pellissier Tanon, Fabian M. Suchanek

▶ **To cite this version:**

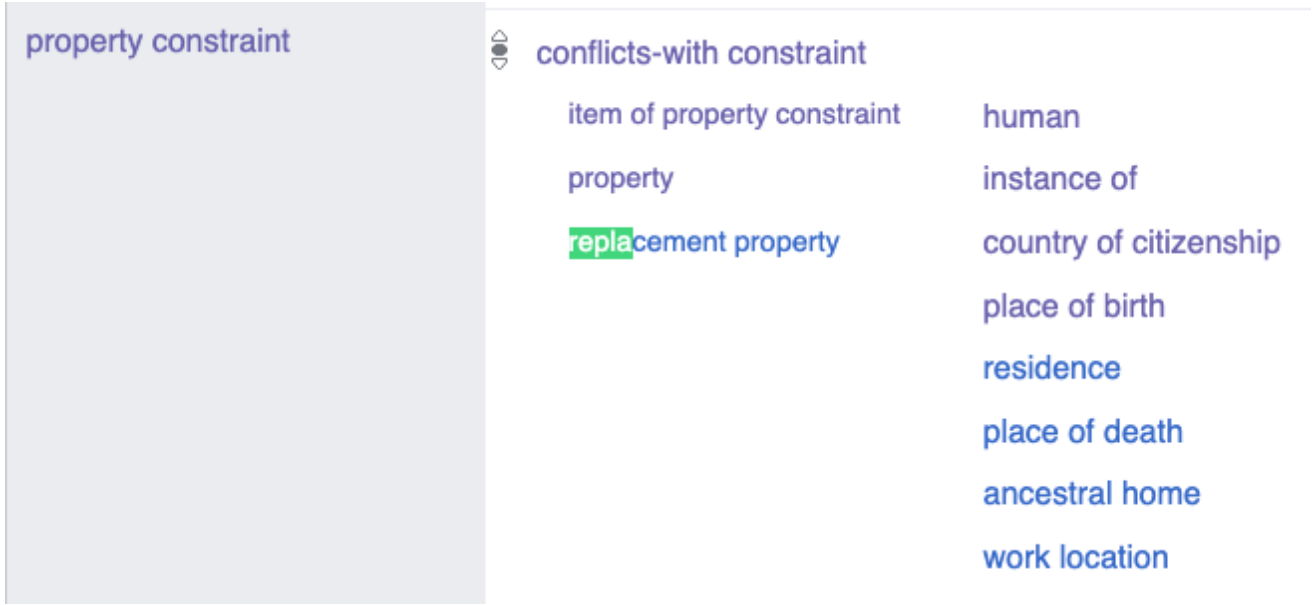
Thomas Pellissier Tanon, Fabian M. Suchanek. Neural Knowledge Base Repairs. European Semantic Web Conference, Jun 2021, Hersonissos (virtual), Greece. pp.287-303, 10.1007/978-3-030-77385-4_17 . hal-03325101

TODO: some slides or notebook to explain how the training data looks like...

Challenge 1: Does an ML approach using this data learn the “right” repairs?

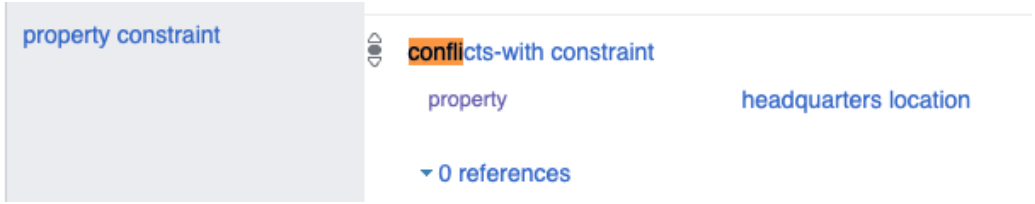
- E.g. some constraints have a “replacement” property “suggestion” or “rule” in case of violations”

- https://www.wikidata.org/wiki/Property:P17



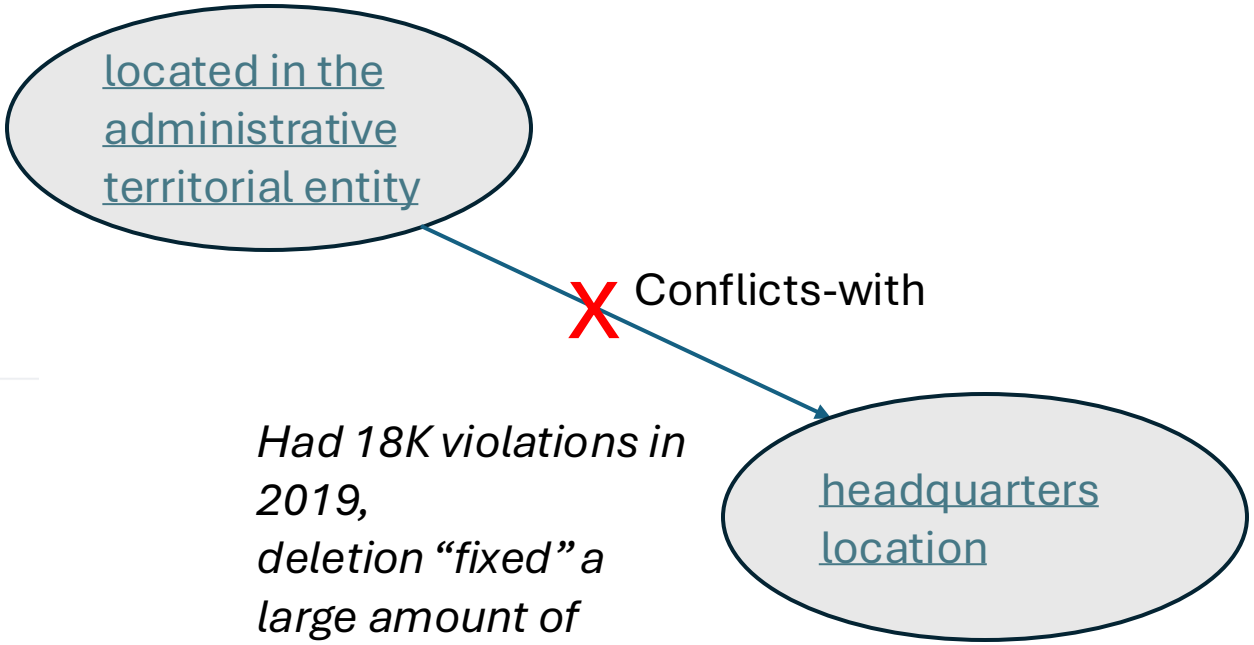
Challenge 2: In KGs, the constraints can also be graphs!

- What if the constraint is changed?
- Can we learn (when it's better to do) constraint repairs?
- What if the constraint is changed?
- We have data also for that! E.g.:
- https://www.wikidata.org/w/index.php?title=Property:P131&oldid=824983023



- BUT:
  - constraint repairs are probably sparse
  - Constraint changes may "trigger" new violations.

2019:

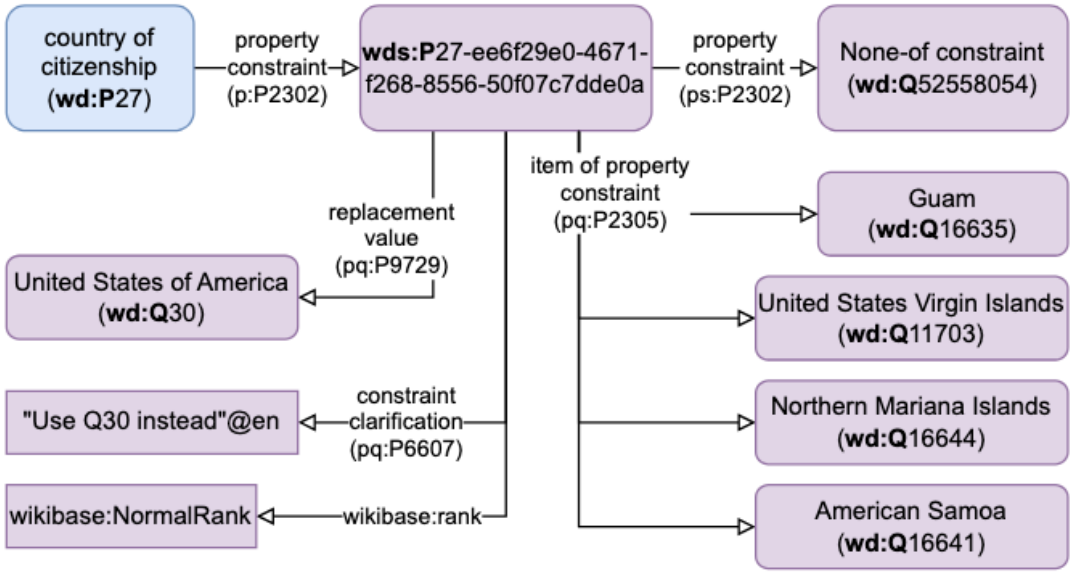


Had 18K violations in 2019, deletion "fixed" a large amount of violations at once!

Challenge 3: Property Graphs: Qualifiers and contextual information – add complexity to repairs

Both instance data and constraints have additional qualifiers, such as ranks, temporal context, etc.:

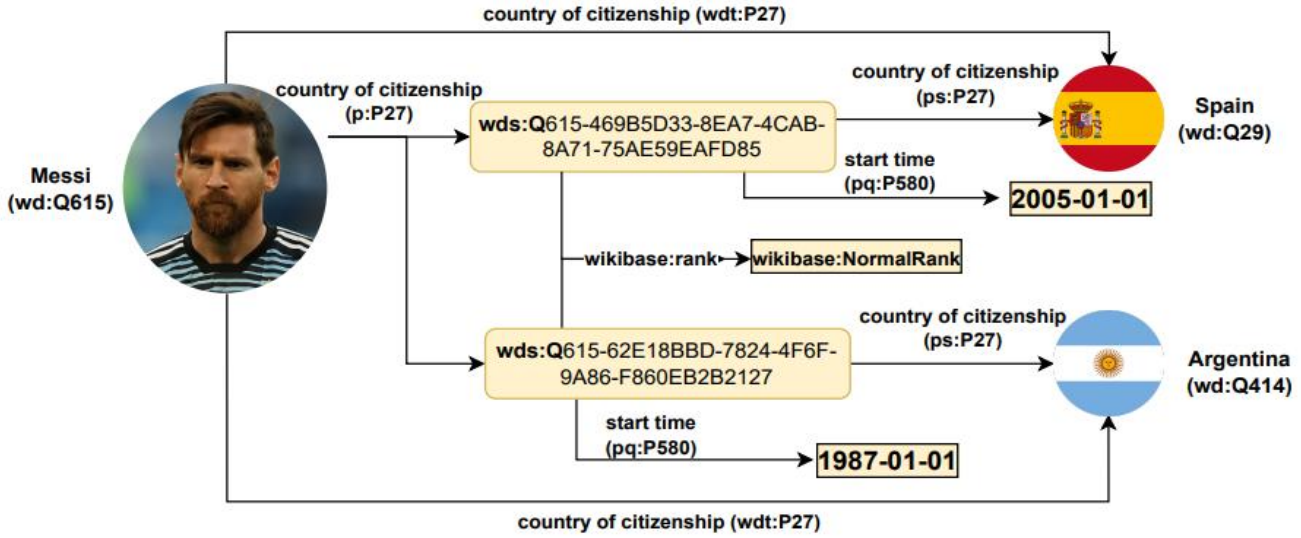
- When should a repair be marked a deprection?
- When should a qualifier be added/removed?
- Can we extend our approach to cover that?



Challenge 3: Property Graphs: Qualifiers and contextual information – add complexity to violation checking and repairs

Example: contextual information in Wikidata instance data:

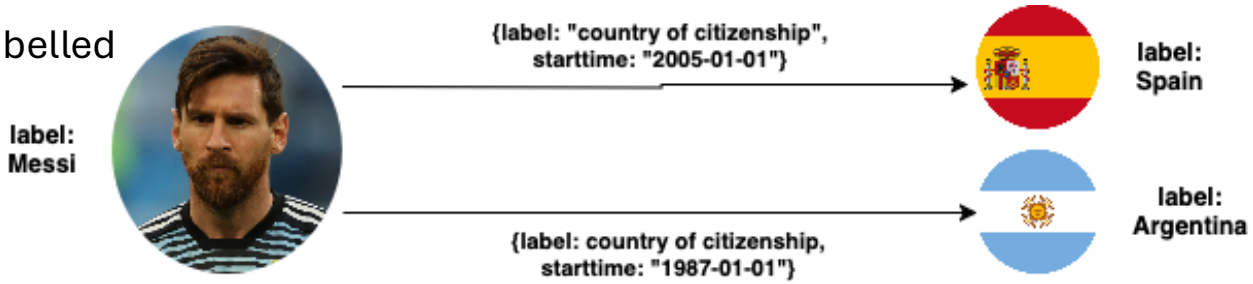
- this is just an ugly "reified" form in RDF…



Some "repairs" could also be done by:

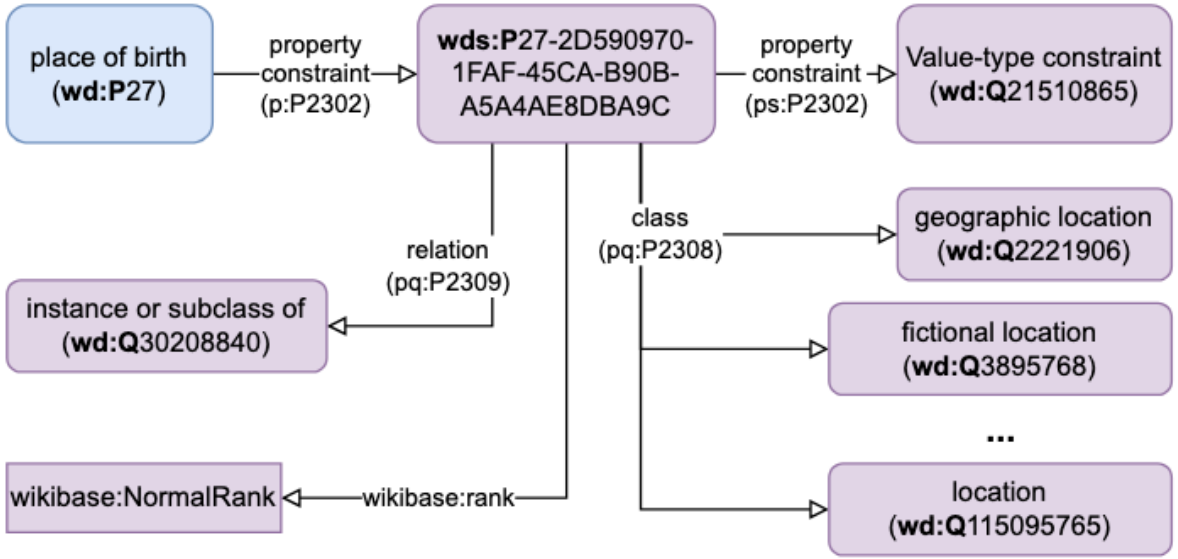
- Changing "ranks" of statements to only one ***wikibase:preferredRank***
- Adding disjoint start and end times

- … of what people often call a Labelled Property Graph (LPG)*:



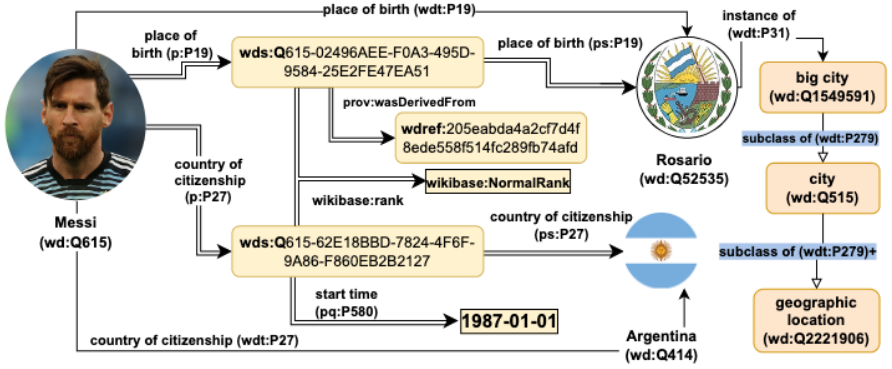
Challenge 3: Property Graphs: Qualifiers and contextual information – add complexity to repairs

Example: contextual information in Wikidata constraint definitions:



Some “repairs” could also be done by:

- Adding “exceptions” to constrains
- Changing the subclassOf hierarchy!



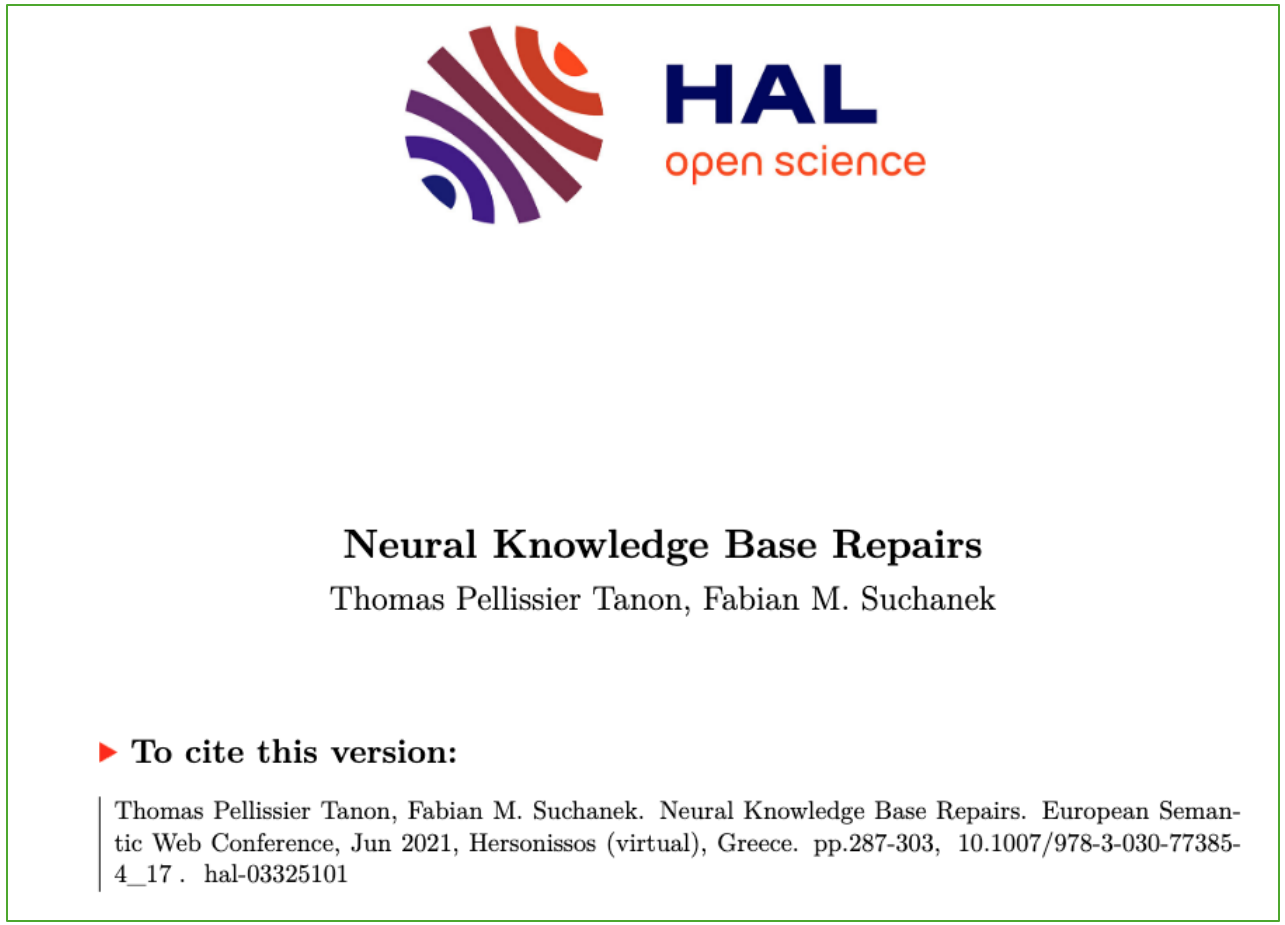
Challenge 4: How could “repairs” interact?

- Can a (constraint) repairs or changes cause new violations?
- Theoretically, yes!
- Does that happen? Yes!

Empirical Observation:

- *the addition of the language of work or name (P407) qualifier to the existing Required-qualifier constraints for official website (P856) and described at URL (P973) properties after 2019 led to over 1.8 million new violations.*

Summary:



HAL open science

Neural Knowledge Base Repairs

Thomas Pellissier Tanon, Fabian M. Suchanek

To cite this version:

Thomas Pellissier Tanon, Fabian M. Suchanek. Neural Knowledge Base Repairs. European Semantic Web Conference, Jun 2021, Hersonissos (virtual), Greece. pp.287-303, 10.1007/978-3-030-77385-4_17 . hal-03325101

- We will try to learn repairs from looking at historical repairs!
- … Extending an existing approach to GNNs
- … We have training data

- Time allowed: we can think about additional challenges, (such as constraint changes)

Group Task:

- Understand the training data
- Try to implement a GNN that learns such repairs, e.g. start with something small, e.g.
  - conflicts with,
  - item-requires-statement,
  - type, Value-type,
  - (single-value)

  constraints
- Try to understand what it can (and can’t?) learn
- ***Compare baseline approach to Tanon et al.***
- ***Think about how to tackle Challenges 1-4***