Wikidata as a Challenge for Rule Systems

Peter F. Patel-Schneider[1], Ege Atacan Doğan[2]

[1]New Jersey, U. S. A.
[2]Istanbul, Turkey

Abstract

Wikidata is a widely used knowledge graph with features that make it desirable for use in challenges for rule systems and subsequent benchmarks. The sheer size of Wikidata is itself a challenge for rule systems. Wikidata's large and broad ontology provides multiple areas for rule-based inference. Wikidata's continual updates provide a challenge for rule systems that retain consequences. Evaluating rule systems against the current data of Wikidata, rather than against a frozen version, prevents lock-in to a particular set of data and diminishing incremental advances against that data. Rule systems that perform well on Wikidata have the possibility of being incorporated into the software supporting Wikidata.

1. Introduction

Current challenges and benchmarks for rule systems are generally based on synthetic data or frozen extracts from existing data sources. We would like to see more evaluation of rule systems on actual, current, independently produced data sources, instead of synthetic data or frozen extracts of data sources. We feel that this kind of evaluation has several advantages that outweigh the disadvantages.

Using actual, current, independently produced data sources eliminates the bias inherent in synthetic data and the extraction process, in both size and data characteristics. It prevents lock-in to the frozen synthetic generation process or the frozen data. It evaluates rule systems on data that they can actually be run on in applications.

We acknowledge that running on actual, current, independently produced data sources adds extra complexity to evaluations. The first evaluation with a data source must provide a process that takes the current version of the data and, if necessary, transform the data into a form that is suitable for evaluation. This process must be able to accept not only the data current at the time of the first evaluation but should also work for later versions of the data. All evaluations much provide a repeatable process for running the system being evaluated not only on the processed data from when the system is first being evaluated but also one that should work on later versions of the data. (Given the difficulties of creating processes that work in the future, this effectively means that evaluation teams should commit to providing help for future evaluations.)

We further propose that an excellent data source for rule systems is Wikidata—not portions of Wikidata but all of Wikidata. Wikidata is a large, complex knowledge graph with a large, complex ontology, but not too large to be impossible to process for current rule systems. Wikidata is being actively edited and is growing.

Wikidata lacks a formal basis and also lacks rules, which might lead one to think that rule-based systems have no place in Wikidata. But the intended meaning of many Wikidata constructs can be formalized as rules in a suitable logic and Wikidata has a large number of constraints that can also be formalized as rules. We feel that this implicit Wikidata suitable for the evaluation of rule-based systems and can form the basis of challenges for rule-based systems.

Wikidata-based evaluations can range from simply answering queries against Wikidata plus some of the rule implicit in Wikidata through completion of Wikidata with some of these rules. As Wikidata

RuleML+RR'25: Companion Proceedings of the 9th International Joint Conference on Rules and Reasoning, September 22–24, 2025, Istanbul, Turkiye
EMAIL: pfpschneider@gmail.com (P. F. Patel-Schneider); egeatacandogan@gmail.com (E. A. Doğan)
ORCID: 0009-0000-7789-7459 (P. F. Patel-Schneider); 0009-0000-0854-2064 (E. A. Doğan)

constantly changes due to edits these evaluations can not only be on a fixed, but current, version of Wikidata but also against Wikidata subject to a current stream of changes, stressing the truth maintenance aspect of rule systems.

Although Wikidata is generally considered to be of good quality, there are errors in Wikidata. Another evaluation task for rule-based systems on Wikidata is finding contradictions in Wikidata and then finding likely causes for these contradictions.

Wikidata currently suffers from not having a mechanism to show the implications of the intended meaning of its constructs. A rule system that can effectively process Wikidata would provide a useful service for anyone wanting to edit or use Wikidata and could end up being an official part of Wikidata. This last benefit can work two ways—not only do rule systems benefit from use on Wikidata but Wikidata can benefit from a better interface that can access the consequences of its data as determined by the rule system. As we are interested in improving Wikidata we are willing to provide assistance to any team that wants to use Wikidata to evaluated rule-based systems.

What follows is a description of Wikidata, some rule sets that implement the intended meaning of some parts of Wikidata, and more description of challenges and evaluations that might be performed using Wikidata.

2. Wikidata

Wikidata (https://wikidata.org) [1] is a large, freely-available knowledge graph built from the contributions of many editors. As of September 2025, Wikidata contains more than 118 million items, such as the item `Q39246` (Richard Feynman),[1] and over 10 thousand properties, such as `P31` (instance of). Part of the information about `Q39246` is shown in Figure 1. Each item has a multi-lingual label and other natural language information plus a collection of statements about the item. A statement can be thought of, with some loss of precision, as a simple binary relationship, e.g. `P31(Q39246,Q5)` or `instance of(Richard Feynman,human)`; or as a more-complex relationship that contains qualifier pairs, e.g., `P735(Q39246,"Richard",P1545,1)` or `given name(Richard Feynman,"Richard",series ordinal,1)`. Statements also have a rank (preferred, regular, or deprecated) and can have references supporting the statement. There were about 1.65 billion statements in Wikidata as of early 2025 (https://en.wikipedia.org/wiki/Wikidata) plus about the name number of facts that are not represented as Wikidata statements, notably natural language labels and descriptions of items in multiple languages.

The native form of Wikidata is a collection of Wikibase (https://www.mediawiki.org/wiki/Wikibase) pages, that are variants of the MediaWiki (https://www.mediawiki.org/wiki/MediaWiki) pages used in Wikipedias. There are translations of Wikidata into RDF, both as weekly dumps (https://dumps.wikimedia.org/wikidatawiki/entities/) and as change logs.[2] The "truthy" versions of these do not contain qualifiers or references for statements and omit deprecated-rank statements and regular-rank statements that contrast with preferred-rank statements. The "full" versions of these represent all the information in Wikidata in RDF using a complex encoding that uses multiple RDF triples for each statement. There are about 20 billion triples in the QLever endpoint (measured using a query in the endpoint itself). The truthy dump is about one-tenth the size. Wikidata received between 300 and 1100 edits per minute as of 2020 (https://wikitech.wikimedia.org/wiki/WMDE/Wikidata/Growth) with each edit potentially changing multiple statements. This rate was expected to remain roughly constant.

Wikidata is just a knowledge graph. There is no formal theory underlying Wikidata. There are no rules nor any built-in inference process in Wikidata. What Wikidata does have is an intended meaning for some of its properties and items. For example, Wikidata has two properties that are core to its ontology: `P31` (instance of) and `P279` (subclass of). These two properties are considered to have their usual meaning, i.e., subclass of is transitive and instances of classes are also instances of their superclasses.

[1] In this paper entities from Wikidata will be often shown using their internal ID along with their English label.

[2] There is currently little public documentation on this interface, even though it is used to drive changes to the QLever SPARQL Wikidata endpoint (https://qlever.cs.uni-freiburg.de/wikidata).

Richard Feynman (Q39246)

Item Discussion Read View history Tools

American theoretical physicist (1918–1988) edit

Richard Phillips Feynman | Ofey | Feynman | Dick Feynman | Richard P. Feynman

In more languages

Statements

instance of human edit

1 reference

+ add value

image edit

RichardFeynman-PaineMansionWoods1984 copyrightTamikoThiel bw.jpg

470 × 541; 117 KB

point in time 1984

1 reference

+ add value

signature edit

Richard Feynman signature.svg

585 × 206; 24 KB

0 references

+ add reference

Figure 1: Wikidata Page for Richard Feynman

Wikidata does not provide any support for this intended meaning. Users wishing to access complete subclass and instance relations generally use a SPARQL Wikidata endpoint and use SPARQL path expressions (`wdt:P279*` and `wdt:P31/wdt:P279*`, respectively). This situation is by no means ideal as users have to know to write this more-complex query, with the greater chance of getting it wrong.

Partly as a result of the lack of a formal basis, there are errors in Wikidata. The presence of these errors have been known for quite some time and there was recently an effort to characterize the errors in the Wikidata ontology [2, 3, 4].

This is not to say that there is nothing in Wikidata that can be used to point out errors in Wikidata. Wikidata does have a constraint system (https://www.wikidata.org/wiki/Help:Property_constraints_portal) and there are tools to evaluate the constraints against the data in Wikidata, producing reports on constraint violations. We have recently developed tools to find violations of disjointness in Wikidata [5] and problems with class order [6]. These tools are not based on any formal description of Wikidata and do not use any logic-based systems but instead are combinations of queries and procedural code.

3. Rules for Wikidata

Given the lack of formal basis described above, how can Wikidata be used as a challenge or benchmark for rule systems?

What a rule system can do for Wikidata is to formalize and implement part of the intended meaning of Wikidata—the meaning of Wikidata that is not just the raw data in Wikidata but encompasses the informal meaning for quite a number of its items and, especially, properties.

3.1. Basic Ontology Rules

Perhaps the simplest use for rules[3] in Wikidata is to implement the intended meaning of `P31` (instance of) and `P279` (subclass of), with subclass being transitive and instances of classes being instances of their superclasses.

$$P279(?a, ?c) \leftarrow P279(?a, ?b), P279(?b?c) \tag{1}$$

$$P31(?i, ?c) \leftarrow P31(?i, ?b), P279(?b?c) \tag{2}$$

These rules correspond to the SPARQL path expressions that Wikidata users employ when querying the Wikidata ontology.

This seems to be too simple for current rule systems. But there are still issues of size. With about 118 million items in Wikidata and 2 or 3 billion triples in truthy Wikidata expressed as RDF or 20 billion triples in full Wikidata expressed as RDF even these simple rules could pose a worthwhile challenge for rule systems. The transitive closure of subclass has about 125 million relationships and there are about 1.5 billion total instance relationships as of September 2025 (determined by SPARQL queries) indicating how many inferences would have to be computed to complete these basic ontology properties in Wikidata.

3.2. Subproperty Rule

A further complication is that the above properties are just regular properties, as opposed to being part of the meta-language of Wikidata. So these properties can be affected by other parts of Wikidata, notably `P1647` (subproperty of). As there are subproperties of `P31` (instance of) and also of `P279` (subclass of), the normally-used queries for instance and subclass are incomplete. (To make ontology SPARQL queries complete for a language that has subproperties is not possible in general.) Even for Wikidata, which only has a few relevant subproperties, the required SPARQL queries for truly complete answers for instance and subclass are quite complex.

[3]We will use a neutral clause syntax for rules throughout this paper.

The rule

$$P1647(?p, ?r) \leftarrow P1647(?p, ?q), P1647(?q, ?r) \tag{3}$$
$$?q(?s, ?o) \leftarrow P1647(?p, ?q), ?p(?s, ?o) \tag{4}$$

is needed to add the intended meaning of subproperty, namely that subproperty is transitive and statements for a subproperty are also statements for its superproperties. The resultant set of rules is no longer quite so simple, as it has a variable in predicate position. But even if the rule system does not allow variables in predicate position, the simple trick of using a `holds` predicate can be employed, resulting in a simple set of rules, albeit with a large number of consequences.

3.3. Class Order

Wikidata includes several classes with intended meaning that can be captured with rules.

One group of such classes are the order classes: `Q104086571` (first-order class), `Q24017414` (second-order class), `Q24017465` (third-order class), and `Q24027474` (fourth-order class). (There are other order-related classes but they are not important here.)

Class order is defined as follows: A class is first-order if all its instances are non-classes. A class is $n+1$st-order if all its instances are nth-order classes. Note that these rules should be taken to not mean all instances in the current knowledge graph but instead as over all potential instances. So it is not possible to compute class order just by looking at the current instances of a class.

But it is possible to determine some axioms and rules for class order using the intended meaning of the order classes above.

$$order(?c, 1) \leftarrow P31(?c, Q104086571) \tag{5}$$
$$order(?c, 2) \leftarrow P31(?c, Q24017414) \tag{6}$$
$$order(?c, 3) \leftarrow P31(?c, Q24017465) \tag{7}$$
$$order(?c, 4) \leftarrow P31(?c, Q24027474) \tag{8}$$
$$order(?c, ?n) \leftarrow order(?d, ?n), P279(?c, ?d) \tag{9}$$
$$order(?i, n-1) \leftarrow order(?c, n), P31(?i, ?c) \tag{10}$$

The rules introduce a new predicate for the order of a class, initially determined by it being an instance of one of the order-inducing classes. Then order of a class is the same as order of any of its superclasses. Also, the order of an item is one less than the order of any class it belongs to.

These rules result in over 14 million first-order classes in Wikidata, over 2 million second-order classes, and over 3 thousand third-order classes. (All these results were computed using SPARQL queries that implement the above rules that were hand-crafted to some special circumstances in Wikidata [6].) Over ninety percent of the third-order classes are also second-order classes and ninety percent of the second-order classes are also first-order classes.

A non-empty class has at most one class order, which can be detected using the following rule:

$$empty(?c) \leftarrow order(?c, n), order(?c, m), n \neq m \tag{11}$$

So class order states that many Wikidata classes should be empty, as nearly all of these classes are not empty in Wikidata the following rule results in many contradictions:

$$\bot \leftarrow P31(?i, ?c), empty(?c) \tag{12}$$

Problems with class order in Wikidata are real and do not just result from incorrect class order statements. An interesting challenge for a rule system would be to determine potential root causes of inconsistency involving class order to help reduce the issues in Wikidata.

There are other inferences related to class order in Wikidata. Counting downward instance chains from a class produces a minimum class order. Showing that an entity is both an instance and a subclass of a class shows that the class cannot have an order. (It may be a variable order class, for example) Combining these with the previous rules produces a theory of class order in Wikidata.

3.4. Property Characteristics

Wikidata properties are entities and can belong to classes just like items can. Some of these classes have intended meanings that are amenable to implementation via rules. These classes include Q18647518 (symmetric property), Q18647515 (transitive property), and Q18647519 (asymmetric property). Currently, these intended meanings are implemented or checked by a combination of non-constraining constraints[4] and external programs (*bots*) that add and modify information in Wikidata.

Wikidata properties can also have properties and some of these properties also have intended meanings that are amenable to implementation via rules. These properties include P1696 (inverse property) and P6609 (transitive over). The former is not complete inverse but instead is a partial non-symmetric inverse, whose intended meaning is captured by the following rule:

$$?q(?b, ?a) \leftarrow P1696(?p, ?q), ?p(?a, ?b) \tag{13}$$

The latter is a kind of retributive characteristic, with rule:

$$?p(?a, ?c) \leftarrow P6609(?p, ?pt), ?p(?a, ?b), ?pt(?b, ?c) \tag{14}$$

Although there are not very many Wikidata properties that belong to these classes or have values for these properties, both P279 (subclass of) and P1647 (subproperty of) are instances of Q18647515 (transitive property) and P31 (instance of) P6609 (transitive over) P279 (subclass of).

Implementing this part of basic ontology reasoning could be a significant challenge for rule systems.

3.5. Disjointness

Wikidata has the property P2738 (disjoint union of), which is a statement on a class that it is the disjoint union of a list of other classes. This information is carried in a special qualifier: P11260 (list item).

As a result, reasoning about disjointness in Wikidata requires information from the full dump of Wikidata. But this qualifier only serves to allow disjointness to be more than a binary relationship so reasoning about disjointness could be done in truthy Wikidata with just this special qualifier added, possibly by translating disjoint union statements and their P11260 qualifiers into an n-ary disjoint union fact, but still using predicate P2738.

With this translation, part of the intended meaning for disjoint unions consists of the following rules:

$$disjoint(?c, ?d) \leftarrow P2738(?s, \ldots, ?c, \ldots, ?d, \ldots) \tag{15}$$
$$disjoint(?c, ?d) \leftarrow disjoint(?d, ?c) \tag{16}$$
$$disjoint(?s, ?d) \leftarrow disjoint(?c, ?d), P279(?s, ?c) \tag{17}$$
$$empty(?e) \leftarrow P279(?e, ?c), P279(?e, ?d), disjoint(?c, ?d) \tag{18}$$

These rules first turn the Wikidata n-ary disjoint union into binary disjoint statements. Then disjoint is stated to be symmetric, subclasses of disjoint classes inherit disjointness, and a class that is the subclass of two disjoint classes is an empty class.

We conducted an analysis of disjointness of Wikidata in 2024 [5] and found that there were over 86 thousand classes that were necessarily empty because of disjointness and nearly 10 million entities that were instances of two disjoint classes. (These disjointness problems were found by running many queries against Wikidata and potentially missed a few problems because the queries used the usual way of determining instance and subclass. A system based on rules would allow complete discovery.) Finding disjointness problems adds more rules and more complexity to a Wikidata challenge or benchmark.

[4]Wikidata constraints mostly just add caution notes to Wikidata pages and produce reports.

3.6. Constraints

As indicated above, Wikidata incorporates constraints (www.wikidata.org/wiki/Help:Property_constraints_portal). A constraint in Wikidata is generally a non-enforcing condition that should be met for relationships in a property.

Most of the constraints are very local, such as the single-value constraint, which checks that each item has only one value for a property, such as the single-value constraint on P22 (father).

Constraints can mostly be checked directly in the Wikidata data structures, as they involve properties that do not have subproperties or have any special intended meaning and only look for simple characteristics of the property and its values on an item.

But complete checking of some constraints involves properties that do have subproperties or special intended meaning, such as the constraints on P31 (instance of). Constraints on these properties are incompletely handled within Wikidata. (Some constraints are handled by external *bots*—programs outside the Wikidata software that are run periodically and either produce constraint violation reports or modify Wikidata in some way to satisfy constraints.)

Wikidata constraints are described using natural language and do not have a formalization in Wikidata, but have been formalized in SHACL by Ferranti *et al* [7]. Expressing them as rules is fairly simple.

As there are many constraints in Wikidata and they cover many relationships, another challenge problem for rules would thus be to determine constraint violations in Wikidata. There are many of these, as the voluminous constraint violations reports (www.wikidata.org/wiki/Wikidata:Database_reports/Constraint_violations) show.[5]

A rule system that could effectively process the constraints of Wikidata would provide a useful service for Wikidata.

3.7. Qualifiers

One of the challenges of Wikidata as far as rules go is how to incorporate qualifiers in general.

Qualifiers can add information to a statement, such as the disjoint components above. But qualifiers can also contextualize, or limit, a statement [8], for example by using temporal qualifiers, including P580 (start time) and P582 (end time). Aljalabout *et al.* [9] categorize the major qualifiers and propose using a many-sorted logic in which to capture their meaning.

There are several challenges for rules here—a different kind of challenge than one for rule systems. First, to come up with a base logic for qualifiers, such as Attributed Description Logics [10], and a syntax for rules there. Second, to either build a theory in that logic or an extension of the logic to handle contextualizing qualifiers. Third, to produce rules that formalize the meaning of important qualifiers in Wikidata.

4. Incorrect Information in Wikidata

As the above discussion indicates, Wikidata has many contradictions or potential contradictions when its intended meaning is considered. Many of these contradictions are hard to detect without some sort of inferencing. Helping to reduce inconsistencies in Wikidata is thus a useful challenge for rule systems.

It can make sense to assume that some parts of a dataset are consistent and suitable for inference. Even though Wikidata data is generally of high quality, and most statements in it are correct, having even a low percentage of incorrect information can result in many inferences that are not correct. For example, one might want to assume that instance information in Wikidata is correct enough to find problems with class order or assume that class order information is correct enough to find problems with instance. But both have errors and it is unclear how to use one sort of information to find problems with the other.

An interesting challenge is thus to better use the information in Wikidata, potentially along with external information, to detect problems and potential fixes. The role of the rule system in this challenge

[5]These reports miss some constraint violations.

is to detect problems, as evidenced by inferred contradictions, and come up with potential causes of the problems. The overall system would provide guidance, by comparing with other good information sources, by using information in Wikidata to estimate information quality, or by other means.

5. Reasoning Modes

There are several modes of operation that a rule system can employ. A rule system can be forward-chaining, where inferences are performed and their consequences added to a knowledge base and retained indefinitely in advance of any requests for information. A rule system can be backward-chaining, where inferences are performed only as necessary to answer requests and not retained after the request has been answered. A rule system can eschew these pure modes by sometimes adding consequences that are not needed to answer requests or remembering consequences of past inferences for future use.

A challenge for forward-chaining rule systems is simply to compute the consequences of part of the intended meaning of Wikidata, using rules that formalize this intended meaning.

For backward-chaining or mixed rule systems a challenge needs to also include a set of queries. Instead of using a fixed set of queries, a challenge or benchmark could instead be based on parameterized queries with parameter values determined by sampling from the current version of Wikidata. For example, a very simple parameterized query would be to determine the full set of types of an item, as in

$$\bot \leftarrow P31(?p, ?t) \tag{19}$$

where *?p* is a parameter, not a variable. Parameter values can be determined by various means such as sampling from the results of SPARQL queries, for example

```
SELECT ?p WHERE { ?p wdt:P31 wd:Q2424752}
```

to sample from `Q2424752` (product).

5.1. Updates

One issue with a rule system for Wikidata is that Wikidata is edited frequently. A rule system for Wikidata that records consequences needs to be able to forget these consequences if their support is removed. As Wikidata is updated many times per minute, responding to edits can in itself be a challenge for a rule system.

So an additional challenge for any rule system that records consequences is to keep up with the edits to Wikidata. At the same time that the rule system is processing edits it also needs to be able to process queries, so the update process needs to not consume too many compute resources.

6. Running a Wikidata Challenge or Benchmark

Wikidata data is available under the CC0 license (creativecommons.org/publicdomain/zero/1.0/), which permits free use for any purpose, so there is no problem in using Wikidata for evaluating rule systems. The current dumps of Wikidata are large but readily available and their format is well documented, as described at www.wikidata.org/wiki/Wikidata:Database_download. These dumps are only retained for a short period of time, so any reuse of historical versions of Wikidata would require placing the dump in a separate repository. The stream of edits to Wikidata was mostly used internally and is only now being made available for external use. Documentation is being prepared for this stream.

Aside from the size of Wikidata, running a challenge or benchmark for living Wikidata would require more work because of the need to keep previous results updated. When a Wikidata-related problem is first proposed, instead of providing a fixed dataset there will have to be a process for constructing the dataset from the current version of Wikidata. Sometimes this might be just obtaining a current RDF or other dump of Wikidata and possibly a current stream of updates but sometimes some pre-processing

might also be needed. For problems that need a set of queries or other accesses, there also needs to be a process for constructing these queries that uses the current version of Wikidata. Finally, there needs to be a process for running the challenge or benchmark that can be repeated by others and with different rule systems. Some of these requirements are already in place to support repeatability, the processes need to work for not just a given dataset and query set but for new versions of Wikidata.

Running the challenge or benchmark later will not only involve creating the current dataset and query set and running it on a new rule system but also updating the results for other rule systems on the current data. As well, a process for running the new rule system on future versions of the data needs to be created.

A challenge or benchmark for updates also needs means to receive the update stream, process updates as necessary, and send the update stream to the rule system. There also needs to be a way of also sending queries to the rule system during updates.

Repeatability can be supported as it always has been by requiring evaluations to include the actual data that they used for their evaluations. But to achieve prominence, any rule system has to perform well not only at a single point but needs to continue to perform well as Wikidata evolves. This prevents lock-in to the single dataset and the common situation of marginal improvements on that dataset.

Our ultimate goal in proposing Wikidata challenges is to improve Wikidata itself, so we would like to evaluate systems based on how well they might support this goal. But that by no means an objective measure, so we must determine which metrics best approximate what the community would deem useful. We thus feel that correctness and the ability to perform the task are the two primary evaluation metrics, making time a secondary metric. On the correctness side, an unsupported inference would be disqualifying. For evaluations involving updates the ability to keep up with the update stream becomes a primary metric.

7. Conclusion

Using a challenge or benchmark based on parts of the intended meaning of living Wikidata has benefits for rule systems.

Wikidata can be viewed in several ways as a challenge or benchmark. Wikidata has over 118 million entities. There is truthy Wikidata, where Wikidata is viewed as only having standard facts, including Wikdiata statements, in a standard Boolean logic. Even in truthy Wikidata there is the need to either allow variables in rules or to encode facts to get around this need. There is full Wikidata, where Wikidata statements have associated qualifiers, requiring the ability to work outside of simple logics, such as a temporal logic, or encoding these extensions in a simple logic. Truthy Wikidata has over 2 billion facts and full Wikidata has a order of magnitude more. The intended meaning of entities in Wikidata give rise to a variety of rulesets, ranging from small rulesets in simple logics, but with large consequence sets, to large rulesets on temporal and other logics. Wikidata is updated hundreds of times per minute. All of these place stresses on performance characteristics of rule systems, so even fixed snapshots of Wikidata are a source of good challenges and benchmarks for rule systems. As Wikidata is continually updated, it can serve as a living challenge or benchmark, preventing lock-in to a fixed dataset.

But the best argument for using living Wikidata as a challenge problem for rule systems is that Wikidata does not provide good support for the intended meaning of even entities that are central to it and its ontology. This places undue burdens on users of Wikidata and creates an opening for systems that can effectively implement this intended meaning. A rule system that can reliably and efficiently handle Wikidata could end up being part of the software that implements Wikidata and thus used and potentially seen by the large community of Wikidata editors.

We would propose three initial challenges involving Wikidata for rule systems. The first challenge is to complete the truthy Wikidata ontology based on the intended meainings of P31 (instance of), P279 (subclass of), and P1647 (subproperty of). This challenge is simple and likely doable by multiple rule systems in reasonable time. The second challenge is to complete the full Wikidata ontology based on at least all the intended meanings above except constraints, and calculating temporal extents for the

inferred statements. This challenge requires rule systems that can handle qualified statements, and may spur development of such systems. These two challenges can be run as both static and updating variants. The third challenge is to find good changes that can fix problems related to disjointness or class order or constraint violations. This last challenge would be judged by members of the Wikidata community.

Declaration on Generative AI

When preparing this paper for publication, one of the authors did numerous Google web searches to try to find out how to include the correct fonts and copyright clause. The results of these searches included results from a generative AI system. Some of these generative AI results were skimmed or read by the author, but none of the information in them was useful.

References

[1] D. Vrandečić, M. Krötzsch, Wikidata: A free collaborative knowledgebase, Communications of the ACM 57 (2014) 78–85.

[2] D. Ammalainen, Wikidata ontology issues: Suggestions for prioritisation based on the perceived frequency of occurrence and the severity of impact on data re-use, commons.wikimedia.org/wiki/File:Wikidata_ontology_issues_%E2%80%94_suggestions_for_prioritisation_2023.pdf, 2023. Accessed 2025-05-12.

[3] M. Abdulai, L. Lacroix, Wikidata: Ontology issues prioritization, www.wikidata.org/wiki/Wikidata:Ontology_issues_prioritization, 2023. Accessed 2025-05-12.

[4] L. Pintscher, Wikidata survey on ontology issues, potential solutions, www.wikidata.org/wiki/Wikidata_talk:Ontology_issues_prioritization#Overview_of_potential_solutions, 2023. Accessed 2025-02-17.

[5] E. A. Doğan, P. F. Patel-Schneider, Disjointness violations in Wikidata, in: Fifth International Knowledge Graph and Semantic Web Conference, 2024. A longer version available as arxiv.org/abs/2410.13707.

[6] P. F. Patel-Schneider, E. A. Doğan, Class order disorder in Wikidata and first fixes, arxiv.org/abs/2411.15550, 2024.

[7] N. Ferranti, A. Polleres, J. F. de Souza, S. Ahmetaj, Formalizing property constraints in wikidata, in: Wikidata'22: Wikidata workshop at ISWSC 2022, 2022.

[8] P. F. Patel-Schneider, Contextualization via qualifiers, in: First International Workshop on Contextualized Knowledge Graphs at ISWC 2018, 2018.

[9] S. Aljalbout, G. Falquet, D. Buchs, Handling Wikidata qualifiers in reasoning, arxiv.org/abs/2304.03375, 2023.

[10] M. Krötzsch, M. Marx, A. Ozaki, V. Thos, Attributed description logics: Ontologies for knowledge graphs, in: The 16th Internaional Semantic Web Conference, 2017.