PG-Schema with Property Constraints validator in Rudof*

Jose-Emilio Labra-Gayo[1,*,†], Dominik Tomaszuk[2,†], Diego Martín-Fernández[1,†], Samuel Bustamante-Larriet[1,†], Álvaro García-Fernández[1,†] and Daniel Fernández-Álvarez[1,†]

[1]University of Oviedo, Spain
[2]University of Bialystok, Poland

Abstract

This paper presents the integration of Property Constraints into PG-Schema, implemented within the rudof system. While rudof was originally designed for RDF data and RDF data shapes — Shape Expressions (ShEx) and Shapes Constraint Language (SHACL) — it has evolved to support other graph models. This includes Labeled Property Graphs (LPGs), which can be used for Knowledge Graphs implementations. Our demonstration showcases the validation of property graph schemas and data using this extension.

Keywords

Labeled Property Graphs, Graph schema, Property Constraints, RDF, Data Quality, Shape Expressions, Data shapes, Knowledge Graphs

1. Introduction

Knowledge graphs are powerful tools for organizing and connecting information, making it easier to discover patterns and insights. They represent data as entities and relationships, enabling applications such as semantic search and data integration. Two main technologies are commonly used for implementing Knowledge Graphs: 1) RDF (Resource Description Framework), which is a popular solution in the semantic web community, offering a flexible and interoperable model based on the use of URIs, and 2) Labeled Property Graphs (LPGs), which allows efficient storage by promoting the use of sets of properties and values in both nodes and edges.

In the case of RDF, two main technologies have been proposed to describe and validate the topology of the graph: Shape Expressions (ShEx) [1] and Shapes Constraint Language (SHACL) [2]. Since their inception, several tools have been implemented that can be used to validate RDF data against a ShEx schema or a shapes graph. One of those tools is rudof [3][1], a Rust implementation of both ShEx and SHACL that aims to support RDF data and related graph data models.

In the case of Property Graphs, there have been several proposals for a schema language. One of those proposals, PG-Schema [4], has been taken as the basis for an extension that adds property constraints [5]. In this demo, we will present the rudof system as well as the PG-Schema PC prototype that has been integrated into rudof, which enables validation of Labeled Property Graphs using PG-Schema with property constraints.

The source code is available and is published as an independent crate [2] enabling its usage as a library by third parties.

Woodstock'22: Symposium on the irreproducible science, June 07–11, 2022, Woodstock, NY

*You can use this document as the template for preparing your publication. We recommend using the latest version of the ceurart style.

*Corresponding author.
labra@uniovi.es (J. Labra-Gayo); d.tomaszuk@uwb.edu.pl (D. Tomaszuk); martinfdiego@uniovi.es (D. Martín-Fernández); bustamantesamuel@uniovi.es (S. Bustamante-Larriet); garciafalvaro@uniovi.es (Á. García-Fernández); fernandezalvdaniel@uniovi.es (D. Fernández-Álvarez)
https://labra.weso.es/ (J. Labra-Gayo)
0000-0001-8907-5348 (J. Labra-Gayo); 0000-0003-1806-067X (D. Tomaszuk); 0009-0003-6640-9474 (D. Martín-Fernández); 0009-0005-8631-2682 (S. Bustamante-Larriet); 0009-0008-9390-6210 (Á. García-Fernández); 0000-0002-8666-7660 (D. Fernández-Álvarez)

[1]https://rudof-project.github.io/
[2]https://crates.io/crates/pgschema

The rudof system publishes binaries in Windows, Linux, and Mac, and offers Python bindings. We provide runnable demos as Jupyter notebooks in rudof's project page[3]

2. PG-Schema validation examples

As an example, Figure 1 represents a simple property graph with information about 2 persons — Alice Cooper and Robert Smith — and a course — Algebra —. Alice knows Robert since 2020, while Robert has been enrolled in the Algebra course between 2024 and 2025.

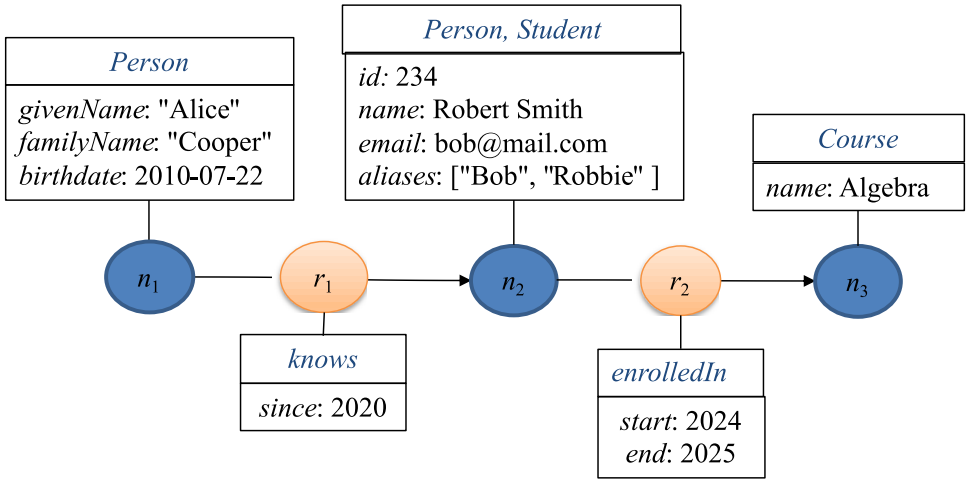

Figure 1: Visual representation of an example property graph

The previous example can be represented following YARS-PG [6] notation as it is shown in Figure 2. In [5], we presented an extension of PG-Schema that adds property constraints, thereby increasing the language's expressiveness. For example, a PG-Schema with property constraints is represented in Figure 3.

The schema declares a node type `PersonType` which describes the content of nodes that represent a `Person` as having either a `name` which must be a `STRING` or the combination of `givenName` plus `familyName`, and an optional `birthdate` which must be of type `DATE`.

It also declares a node type `StudentType` which inherits the content of `PersonType`. This means that nodes that conform to `StudentType` must also conform to `PersonType` and to the new content model. In this case, this means that it must also have the label `Student`, and three properties: `id`, which can be either a `STRING` or an `INTEGER`; an email, which must be a `STRING` conforming to a regular expression that checks that there is an `@`; and zero or more aliases of type `STRING`.

Nodes of type `CourseType` must have a label `Course` and two properties: `name`, of type `STRING`, and `credits` of type `INTEGER`. The number of credits should also be greater than `3`.

Finally, the PG-Schema also declares two edge types for the relationships. An edge type `KnowsType` is declared with the label `knows` and one property `since` of type `INTEGER`. Another edge type `EnrolledInType` is declared with the label `enrolledIn` and two properties: `start` and `end`, both of type `INTEGER`.

The rudof system has been extended with support for Property graphs and PG-Schema. It enables to check that both a property graph and a PG-Schema are syntactically valid, and then to validate that the nodes in a property graph conform to a given PG-Schema. In order to do the validation, the system takes as input a map from nodes/edges to node/edge types like the one that appears in Figure 4.

The validation in rudof can be run as described in Figure 5. In this case, the output shows that the corresponding nodes/edges have been validated and conform to the specified types, including an explanation.

[3] https://rudof-project.github.io/tutorials/

```
(n1 {Person} [ givenName: "Alice", familyName: "Cooper", birthdate: "2010-07-22" ])
(n2 {Person, Student} [ id: 234, name: "Robert Smith", aliases: ["Bob", "Robbie"], email: "bob@mail.com" ])
(n3 {Course} [ name: "Algebra", credits: 6 ])
(n1) - (e1 {knows} [since: 2020 ]) -> (n2)
(n2) - (e2 {enrolledIn} [start: 2024, end: 2025 ]) -> (n3)
```

Figure 2: Property graph example in YARS-PG (demo.pg)

```
CREATE NODE TYPE ( PersonType : Person {
    name: STRING ||
    ( givenName: STRING,
      familyName: STRING
    ),
    OPTIONAL birthdate: DATE
});

CREATE NODE TYPE ( StudentType : @PersonType & Student {
    id: INTEGER | STRING,
    email: STRING CHECK REGEX "^[\w.-]+@[\w.-]+\.\w+$",
    aliases: STRING *
});

CREATE NODE TYPE ( CourseType: Course {
    name: STRING,
    credits: INTEGER CHECK (> 3)
});

CREATE EDGE TYPE (:PersonType)
                 -[KnowsType : knows { since: INTEGER }]->
                 (:PersonType);

CREATE EDGE TYPE (:StudentType)
                 -[EnrolledInType : enrolledIn { start: INTEGER, end: INTEGER }]->
                 (:CourseType);
```

Figure 3: PG-Schema with property constraints (demo.pgs)

```
n1:PersonType, n2:StudentType, n3:CourseType, e1:KnowsType, e2:EnrolledInType
```

Figure 4: Validation map (demo.map)

3. Conclusions

This paper demonstrated rudof, a system that has evolved from a specialized RDF validation library supporting ShEx and SHACL into a framework for multi-model knowledge graphs. By extending support to Property Graphs and introducing a constraint-enabled version of PG-Schema, rudof provides a unified solution for ensuring data quality across both RDF and Labeled Property Graph paradigms.

Regarding future work, we intend to build upon rudof's existing support for RDF 1.2. By leveraging RDF 1.2's ability to handle *'statements about statements'*, the tool currently bridges the gap between the RDF data model and Labeled Property Graphs (LPGs). Our next steps involve extending the ShEx and SHACL validation languages to fully accommodate these new RDF features. Furthermore, we are pursuing a unified schema language for graph data models, operationalizing the framework proposed in [7]. Beyond its core validation capabilities, rudof now includes a Model Context Protocol (MCP) implementation. This feature will enable integrating knowledge graph operations into LLM workflows, allowing users to query data and validate schemas via natural language interfaces.

```
$ rudof validate --mode pgschema --schema demo.pgs --shapemap demo.map demo.pg
n1:PersonType: true  Details:   Labels match Person and record: {birthdate: 2010-07-22, name: Alice}
    conforms to {birthdate: Date(1), name: String(1)}
n2:StudentType: true  Details:   Labels match Student, Person and record: {aliases: [Bob, Robbie], email:
    bob@mail.com, id: 234, name: Robert Smith} conforms to {aliases: String(0..*), email: (String(1)
    Condition((REGEX(^[\w.-]+@[\w.-]+\.\w+$))), id: (Integer(1)  String(1)), name: String(1)}
n3:CourseType: true  Details:   Labels match Course and record: {credits: 6, name: Algebra} conforms to {
    credits: (Integer(1)  Condition((> 3))), name: String(1)}
. . .
```

Figure 5: Example of validation using rudof

References

[1] E. Prud'hommeaux, J. E. Labra Gayo, H. Solbrig, Shape expressions: an RDF validation and transformation language, in: International Conference on Semantic Systems (SEM), ACM, 2014, pp. 32–40. URL: https://doi.org/10.1145/2660517.2660523. doi:10.1145/2660517.2660523.

[2] H. Knublauch, D. Kontokostas, Shapes Constraint Language (SHACL), W3C Recommendation, W3C, 2017. URL: https://www.w3.org/TR/2017/REC-shacl-20170720/.

[3] J. E. Labra-Gayo, A. Iglesias-Prestamo, D. Martin-Fernández, M.-A. Arnaud, rudof: A rust library for handling rdf data models and shapes, in: Proceedings of the International Semantic Web Conference (ISWC24) – Posters and Demos, CEUR, 2024. URL: https://rudof-project.github.io/.

[4] R. Angles, et al., PG-Schema: Schemas for property graphs, Proc. ACM Manag. Data 1 (2023). URL: https://doi.org/10.1145/3589778. doi:10.1145/3589778.

[5] D. Tomaszuk, J. E. Labra Gayo, On property constraints in PG-schema, in: Proceedings of the Knowledge Capture Conference 2025, ACM, New York, NY, USA, 2025, pp. 69–73.

[6] Ł. Szeremeta, D. Tomaszuk, R. Angles, YARS-PG: Property graphs representation for publication and exchange, IEEE Access 12 (2024) 73386–73399.

[7] S. Ahmetaj, I. Boneva, J. Hidders, K. Hose, M. Jakubowski, J. E. Labra-Gayo, W. Martens, F. Mogavero, F. Murlak, C. Okulmus, A. Polleres, O. Savkovic, M. Simkus, D. Tomaszuk, Common foundations for SHACL, ShEx, and PG-schema (2025). arXiv:2502.01295.