Transforming RDF Graphs to Property Graphs using Standardized Schemas

Kashif Rabbani, Matteo Lissandrini, Angela Bonifati, Katja Hose

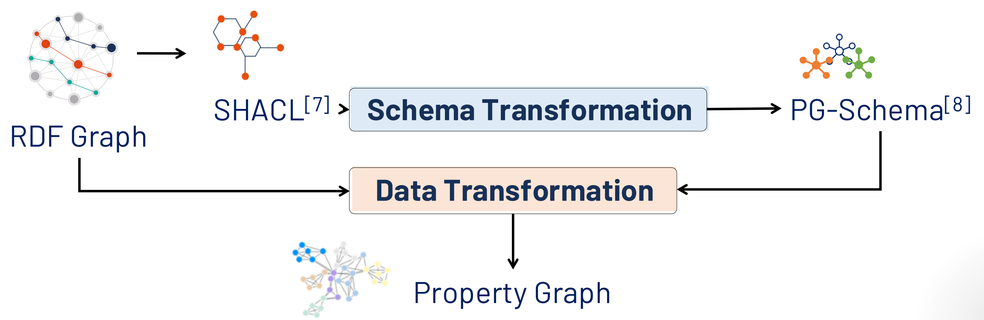
AALBORG UNIVERSITY · UNIVERSITÀ di VERONA · Lyon 1 · TECHNISCHE UNIVERSITÄT WIEN

A **lossless**, **schema-aware**, **fully monotonic** transformation called **S3PG** converts SHACL-constrained **RDF data** into **property graphs**, preserving every fact and constraint.

- How to map RDF triples — including mixed literal/resource values into a property graph data model without **losing information**?
- How to ensure complete **query equivalence** by carrying SHACL cardinalities, datatypes, and hierarchies over to **PG-Schema**?
- How to keep the target graph **incrementally** updated in a **monotonic** way, avoiding costly full reloads as the KG evolves?

You can automatically convert your RDF graph to a property graph without losing any data or breaking any constraints!

Standardized SHACL Shapes-based PG transformation (S3PG)



Interoperability in Knowledge Graphs

Interoperability between RDF & PG models remains unresolved [1]

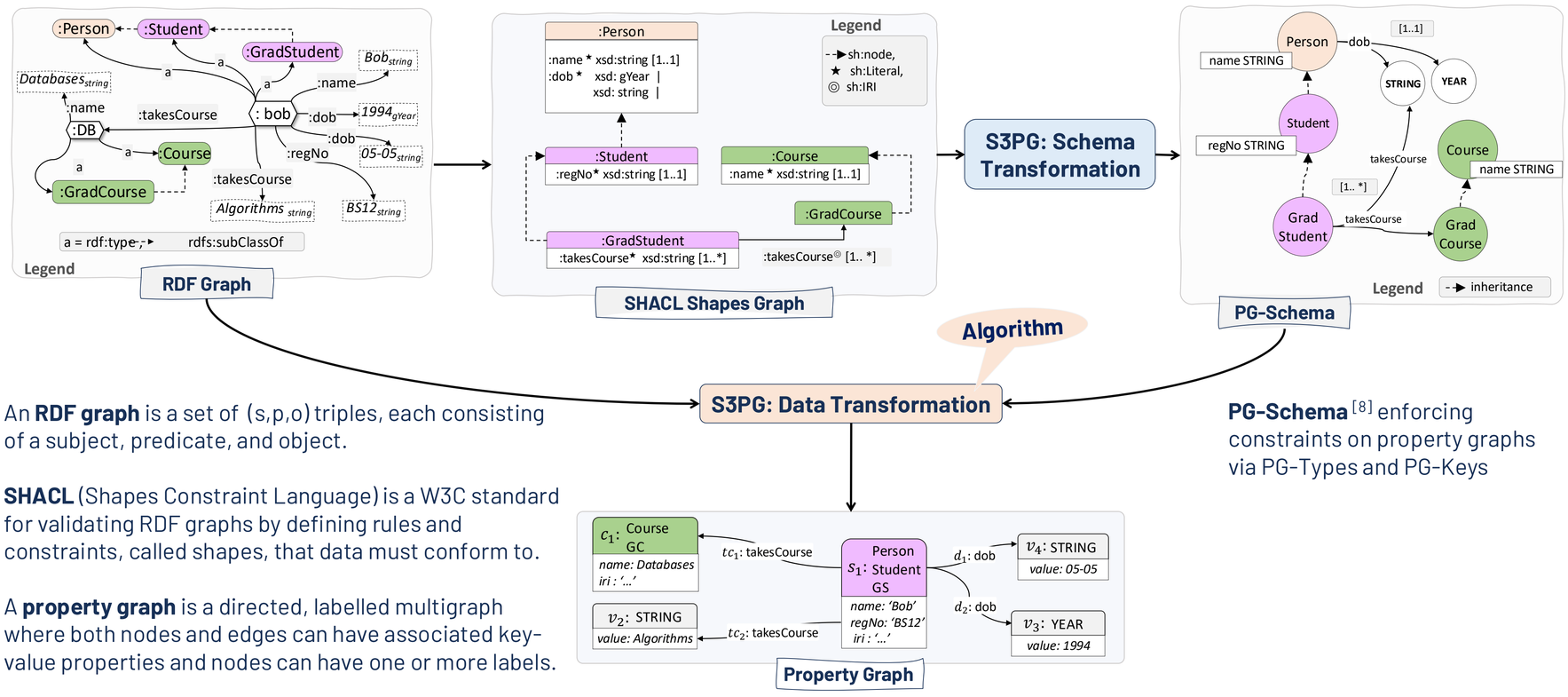
Two main limitations in existing approaches [2,3,4,5,6]

1. **Losslessness**
   Naive mappings drop data or schema constraints
   Up to 70% loss of query results!
2. **Monotonicity**
   Existing non-monotonic methods require rebuilding the entire graph from scratch whenever the data changes.

Goal:
Lossless, schema-aware transformation of RDF graph data into property graph data.

Challenges:
Structural mismatch, heterogenous property types, integrity constraints, scalability, evolving data

S3PG Schema & Data Transformation by Example



An **RDF graph** is a set of (s,p,o) triples, each consisting of a subject, predicate, and object.

SHACL (Shapes Constraint Language) is a W3C standard for validating RDF graphs by defining rules and constraints, called shapes, that data must conform to.

A **property graph** is a directed, labelled multigraph where both nodes and edges can have associated key-value properties and nodes can have one or more labels.

PG-Schema [8] enforcing constraints on property graphs via PG-Types and PG-Keys

Transformation Properties [9]

1. Information Preservation
2. Semantics Preservation
3. Query Preservation
4. Monotonicity

S3PG in a Nutshell: Schema ↔ Data

Schema Transformation Phase (SHACL ⇒ PG-Schema)
Each **NodeShape** becomes a PG-Schema **node type** with the same label(s) and inheritance hierarchy, while every **PropertyShape** is translated into either
(i) a property key on that node type (for single-valued literals), or
(ii) an edge type that records source/target node types, allowed datatypes, and min/max cardinalities.
Union value types, list constraints, and mandatory properties are captured verbatim, yielding a stand-alone PG-Schema that mirrors all SHACL semantics.

Data-Transformation Phase (RDF Triples ⇒ Property Graph)
S3PG reads through the RDF triples only once.
For every resource, it creates a single node, aggregating all *rdf:type* triples into multi-labels.
Non-type triples are processed as follows:
if the object is another resource, an edge of the predicate's label is added;
if the object is a literal, S3PG either stores it as an inline property or, when the schema allows heterogeneous or multi-valued data, as a dedicated value node linked by an edge.
Cardinality and datatype rules are enforced on the fly,
Inserts or deletes can be replayed incrementally, ensuring the property graph remains a monotonic, lossless reflection of its evolving RDF source.

Experimental Evaluation

Machine
Ubuntu 18.04, 16 cores,
1TB HDD, 256GB RAM

Tested on huge datasets!!!

Datasets
DBpedia 2020: 52 Million Triples
DBpedia 2022: 333 Million Triples
Bio2RDF CT: 132 Million Triples

Comparison to
- NeoSemantics [6]
- Rdf2pg [2]

as existing state-of-the-art transformation approaches

Evaluation metrics
- Transformation time
- Accuracy analysis
- Query runtime – each query 10x, 1000 warmup queries

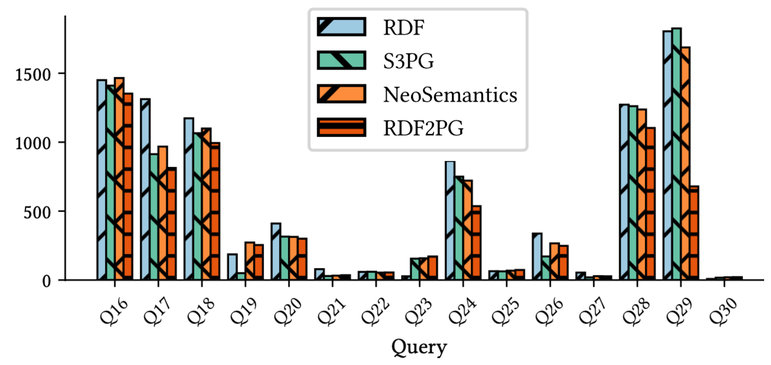
1. Transformation (T) & Loading (L) Time Analysis

	DBpedia 2020			DBpedia2022			Bio2RDF CT		
	T	L	*Sum*	T	L	*Sum*	T	L	*Sum*
S3PG	3 m	9 m	*12 m*	34 m	1.2 h	*1.7 h*	16 m	26 m	*42 m*
rdf2pg	15 m	8 m	*23 m*	1.6 h	58 m	*2.6 h*	45 m	21 m	*1.1 h*
NeoSem	-	-	*38 m*	-	-	*5.5 h*	-	-	*1.3 h*

m: minutes, h: hours

S3PG is 30-70% faster

2. Query Runtime Analysis on DBpedia2022 for Multi Type Heterogenous (Literals & Non-Literals) Queries



3. Accuracy Analysis in Percentages for RDF & PG

		# of GT	S3PG	NeoSem	rdf2pg			# of GT	S3PG	NeoSem	rdf2pg
Single Type	Q1	1,200,712	100%	100%	100%	MT-Hetero (L+NL)	Q16	210,003	100%	99.97%	45.08%
	Q2	282,358	100%	100%	99.46%		Q17	98,595	100%	99.78%	78.66%
	Q3	89,880	100%	100%	99.45%		Q18	93,586	100%	99.99%	79.06%
	Q4	80129.00	100%	100%	99.49%		Q19	1,603	100%	99.69%	99.06%
	Q5	10.00	100%	100%	100%		Q20	11,969	100%	97.76%	91.15%
MT-Homo (L)	Q6	22,566	100%	100%	99.06%		Q21	924	100%	90.48%	79.55%
	Q7	5,509	100%	100%	99.07%		**Q22**	1,831	100%	92.63%	89.30%
	Q8	13	100%	100%	84.62%		Q23	5	100%	100%	100%
	Q9	3	100%	100%	100%		Q24	48,146	100%	94.09%	91.02%
	Q10	52	100%	100%	x		Q25	376	100%	98.14%	94.95%
MT- Homo (NL)	Q11	1,439,679	100%	100%	100%		Q26	13,628	100%	97.23%	87.40%
	Q12	13,111	100%	100%	100%		Q27	687	100%	99.85%	99.27%
	Q13	318,414	100%	100%	100%		Q28	141,570	100%	99.72%	98.88%
	Q14	11	100%	100%	100%		Q29	31,123	100%	99.99%	30.22%
	Q15	55	100%	100%	100%		Q30	7	100%	100%	57.14%

MT: Multi Type, L: Literal, NL: Non-literal

1. Angles, Renzo, Harsh Thakkar, and Dominik Tomaszuk. "RDF and Property Graphs Interoperability: Status and Issues" AMW (2019).
2. Renzo Angles, Harsh Thakkar, and Dominik Tomaszuk. "Mapping RDF Databases to Property Graph Databases" IEEE Access 8 (2020), 86091-86110.
3. Hirokazu Chiba, Ryota Yamanaka, and Shota Matsumoto. "G2GML: Graph to graph mapping language for bridging RDF and property graphs" ISWC (2020), 160-175.
4. E. Haihong, Penghao Han, and Meina Song. "Transforming RDF to Property Graph in Hugegraph" ICEMIS (2020), 1-6.
5. Davide Di Pierro, Stefano Ferilli, and Domenico Redavid. "LPG-Based Knowledge Graphs: A Survey, a Proposal and Current Trends" Information 14(3), 154 (2023).
6. Neo4j. "neosemantics (n10s): Neo4j RDF & Semantics toolkit" https://neo4j.com/labs/neosemantics/
7. SHACL W3C Schema. https://www.w3.org/TR/shacl/
8. Angles, R., Bonifati, A., Dumbrava, S., Fletcher, G., Green, A., Hidders, J., ... & Zivkovic, D. "PG-Schema: Schemas for property graphs" SIGMOD (2023), 1-25.
9. Sequeda, Juan F., Marcelo Arenas, and Daniel P. Miranker. "On directly mapping relational databases to RDF and OWL" World Wide Web (2012), 649-658.
10. Kashif Rabbani, Matteo Lissandrini, Katja Hose. "Extraction of Validating Shapes from very large Knowledge Graphs" PVLDB (2023), 1023-1032.

This research was partially funded by the Independent Research Fund Denmark (DFF) under grant agreement no. DFF-8048-00051B, Agence Nationale de la Recherche VeriGraph (ANR-21- CE48-0015), and the Poul Due Jensens Fond (Grundfos Foundation).

POUL DUE JENSEN GRUNDFOS FOUNDATION · DANMARKS FRIE FORSKNINGSFOND INDEPENDENT RESEARCH FUND DENMARK · anr agence nationale de la recherche

Transforming RDF Graphs to Property Graphs using Standardized Schemas
In Proceedings of the ACM on Management of Data, Volume 2, Issue 6, Article No. 242, Pages 1 - 25