The Quest for Schemas in Graph Databases

Angela Bonifati[1], Stefania Dumbrava[2,3], Emile Martinez[4], Nicolas Mir[2]

[1]Lyon 1 University & LIRIS CNRS, France
[2]ENSIIE & [3]Institut Polytechnique de Paris, France
[4]ENS de Lyon

Property Graph Schemas: State of Affairs

- Ongoing PG Schema standardisation process (ISO SC32/ WG3) in collaboration with PGSWG [a].
- Early proposal for a concise DDL for Cypher with Neo4j folks along with mechanisms for schema validation and evolution[b]
- While waiting for a standard PG schema, we need mechanisms for schema discovery from property graph instances (focus of my talk).

---

[a] https://ldbcouncil.org/gql-community/pgswg/

[b] Angela Bonifati et al. "Schema Validation and Evolution for Graph Databases". In: *Conceptual Modeling - 38th International Conference, ER 2019*. Springer, 2019, pp. 448–456.

Schema Discovery

Interconnected Data:

- *ubiquitous* (Semantic Web, social networks, scientific repositories,...), *heterogeneous* & *semi-structured*.

Graph Databases:

- NoSQL store for efficiently storing & processing *graph-shaped data*.
- No a priori schema constraints $\rightarrow$ error-prone data integration
- Underlying *property graph model*
  (labeled multigraph with key/value lists attached to nodes & edges)
  $\rightarrow$ rich formalism amenable to *schema discovery*

Schema Discovery for Property Graphs

- Existing schema inference mechanisms are basic:
  - no hierarchies,
  - no complex types.
- Recent work on schema inference using MapReduce (MRSchema)[a]:
  - only considers either node labels or node properties $\rightarrow$ trade-off
  - property co-occurrence *information loss* (label-oriented approach) vs. *extraneous type inference* (property-oriented approach).

[a] Hanâ Lbath, Angela Bonifati, and Russ Harmer. "Schema Inference for Property Graphs". In: *EDBT*. 2021, pp. 499–504.

Overview of DiscoPG's Algorithms

Static Case: discover the schema of a static graph dataset $\mathcal{G}$.

- `GMM-S`: novel *hierarchical clustering algorithm*.
- Based on fitting a Gaussian Mixture Model (GMM).
- Accounts for both node label & property information.

Dynamic Case: update the schema of $\mathcal{G}$ upon modifications.

- `I-GMM-D`: incremental approach; reuses `GMM-S`'s clustering.
- `GMM-D`: recomputation approach; memoization-based `GMM-S`.

Property Graph Model

A **property graph**[a] $\mathcal{G}$ is a tuple $(\mathcal{V}, \mathcal{E}, \rho, \lambda, \sigma)$, where:

- $\mathcal{V}$ and $\mathcal{E}$: disjoint finite sets of vertices, and edges,
- $\rho : \mathcal{E} \rightarrow (\mathcal{V} \times \mathcal{V})$: associates each edge with a pair of nodes,
- $\lambda : (\mathcal{V} \cup \mathcal{E}) \rightarrow \mathcal{P}(\mathcal{L})$: associates a vertex/edge with a set of *labels*,
- $\sigma : (\mathcal{V} \cup \mathcal{E}) \times \mathcal{K} \rightarrow \mathcal{P}(\mathcal{N})$: associates vertex/edges with *properties* and, for each property, assigning a set of values from $\mathcal{D}$.

[a] Renzo Angles. "The Property Graph Database Model". In: *AMW*. vol. 2100. CEUR Workshop Proceedings. 2018.

Property Graph Schemas

Base Types ($\mathcal{BT}$): set of *element types* (L, K, O, E_b), where:

- $L \in \mathcal{L}$: set of labels,
- $K \in \mathcal{K}$: set of property names,
- $O \subseteq \mathcal{K}$: subset of optional property names,
- $E_b \subset \mathcal{BT}$: set of element types b extends.

Example:

```
{'Post': {
'creationDate': 2015-06-24T12:50:35.556+01:002,
'locationIP': 42, 'browser' : 'Chrome',
'length' : 10, 'language' : 'lat.',
'content' : 'Lorem ipsum'}}
```

LDBC `Post` node instance

Base type: $(\{Post\}, K, \{language, content\}, \emptyset)$,
where $K = \{creationDate, locationIP, browser, length\}$.

LDBC Ground-Truth Property Graph Schema

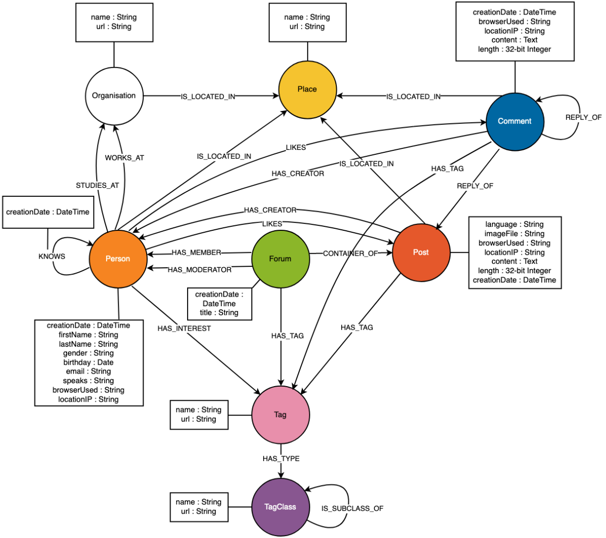


Figure 1: LDBC Property Graph

GMMSchema Methodology

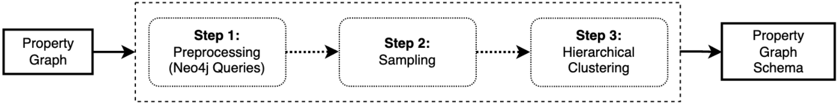


Figure 2: System Workflow

Idea:

- Gaussian Mixture Model (GMM)[a] to discover hierarchical node types.
- for every node label, run GMM algorithm to fit a mixture of normal distributions & use the resulting model for clustering.
- re-iterate procedure in each sub-cluster.

[a] Arthur Dempster and et al. "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *Royal Statistical Society J.* 39 (1977), pp. 1–22.

GMMSchema Base Algorithm (`GMM-S`)

- Collect node labels $\mathcal{L}_{\mathcal{G}}$ & their number of occurrences.
- For each label $L \in \mathcal{L}_{\mathcal{G}}$ (in descending frequency order), iteratively process the set C of all nodes with label L.
- *Reference Base Type* (b_{ref}): most general type for C
  $\rightarrow$ built at each step from all of its node labels
  $\rightarrow$ accounts for the most frequent properties.
- *Feature vector*: constructed from the similarity scores of all nodes in C w.r.t b_{ref} & used to fit a GMM model.
- *EM algorithm*: parameter estimation for Gaussian mixture
  $\rightarrow$ discovered node types.
- *Hierarchical clustering ($\mathcal{C}_{\mathcal{H}}$)*: update b_{ref} with overlapping properties, record C sub-clusters & recursive call in each.

GMMSchema Example

Illustrating the discovery of the sub-types for `Post`-labeled nodes:

- **Parent Node Base Type**: $b = (\{Post\}, K, \emptyset, \emptyset)$,
  where $K = \{creationDate, locationIP, browser, length\}$.
- Run GMM; the new reference nodes are:
  $b_1 = (\{Post\}, K, \{language, content\}, \{b\})$ and
  $b_2 = (\{Post\}, K, \{imageFile\}, \{b\})$
- Repeating the procedure in each sub-cluster does not infer new types, as all nodes in each share the same properties.
  $\rightarrow$ **new discovered sub-types**: $Post1$ and $Post2$.

Discovered LDBC Property Graph Schema

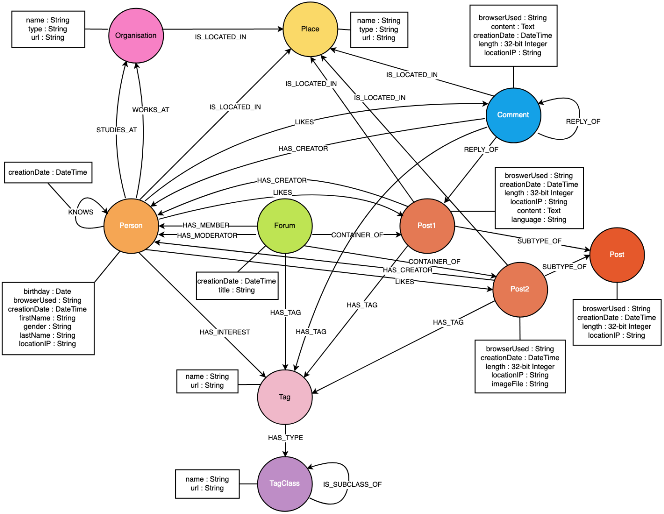


Figure 3: LDBC Property Graph GMMSchema

Experimental Evaluation: Schema Quality (I/II)

Dataset	Nodes	Edges	Node Labels	Edge Labels	Unlabeled
LDBC	1577397	8179418	7	14	0
Mb6	486267	961571	10	3	0
Fib25	802473	1625428	10	3	0
Covid19	36025729	59768373	121	168	474

Figure 4: Dataset characteristics prior to schema discovery.

Dataset	Node Types	Edge Types	Subtype Edges	Hierarchy Depth
LDBC	17	36	9	2
Mb6	19	27	14	4
Fib25	26	106	21	6

Figure 5: Dataset characteristics with GMMSchema discovery.

Experimental Evaluation: Schema Quality (II/II)

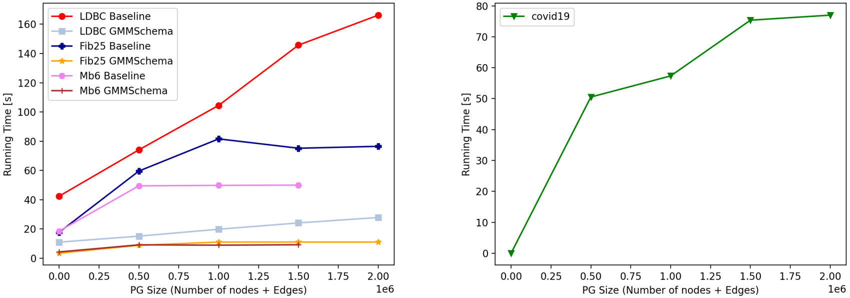
- 2-3 discovered types/label & 3 orders of magnitude more edge types.
- MRSchema infers up to 3 times more node types, up to 3 orders of magnitude more edge types, up to 7 orders of magnitude more subtype edges (for mb6) → up to double the hierarchy depth.

Dataset	Rand Index	AMI	Precision	Recall	F1-score
LDBC	0,96	0,91	1,0	1,0	1,0
Mb6	0,79	0,49	1,0	1,0	1,0
Fib25	0,75	0,41	1,0	1,0	1,0
Covid19	0,94	0,71	NaN	NaN	NaN

Figure 6: GMMSchema clustering quality estimates.

- However, most MRSchema inferred nodes are spurious.
- GMMSchema: perfect accuracy by also leveraging node labels.

Experimental Evaluation: GMMSchema Runtime



(a) LDBC, Fib25, Mb6 **(b)** Covid19

Figure 7: GMMSchema vs. MRSchema total avg. runtimes.

$\rightarrow$ GMMSchema speeds-up schema discovery:
$\times$ 5 (for Mb6) & $\times$ 8 (for LDBC and Fib25).

Inputs:

- Discovered schema for $\mathcal{G}$, as computed by `GMM-S`.
- Graph updates Δ: set of nodes inserted into $\mathcal{G}$.

For each node in Δ:

- Compute its similarity score w.r.t every reference base type corresponding to the clustering $\mathcal{C}_{\mathcal{H}}$.
- Assign it to the cluster maximizing this similarity.

Performance only depends $|\Delta|$ & $|\mathcal{C}_{\mathcal{H}}|$:
$\rightarrow$ *multiple efficient iterations*

$\rightarrow$ *highly robust in practice* (maintains schema quality).

GMM-D

Inputs:

- Discovered schema for $\mathcal{G}$, as computed by `GMM-S`.
- Graph updates Δ: set of nodes inserted into $\mathcal{G}$.

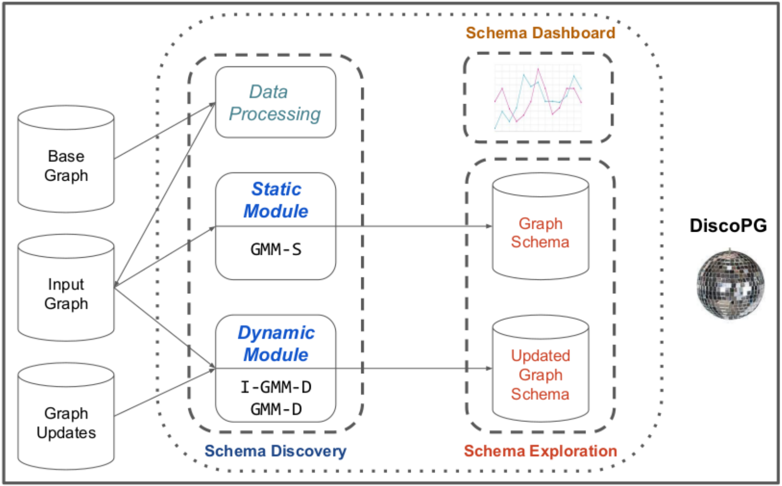
Process the *updated graph* using `GMM-S`, optimized to:

- track the sub-clusters unchanged by the classification step. (no nodes assigned, due to reference base type dissimilarity)
- memoize & avoid recursive calls in these sub-clusters.

W.r.t `I-GMM-D`:
$\nearrow$ *convergence*, $\nearrow$ *iteration-wise runtime*, $\searrow$ *robustness*.

Trade-off: performance vs. quality

DiscoPG System[1] – Workflow Diagram



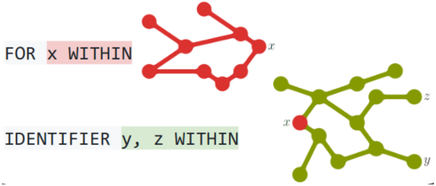
[1] Accepted in VLDB 2022 (demo track)

Conclusions

- DiscoPG: first schema discovery approach for property graphs (accounting for *both* node labels & properties).
- addresses previous limitations (incomplete/spurious node inference) while showing superior accuracy & performance.
- promise of employing statistical methods for schema discovery.
- extensibility to future standard PG schema languages

Perspectives

- integrating topological information (graph embeddings),
- extension to streaming graphs,
- discovery of property graph constraints (PG-Keys, ...)



PG-Keys: Keys for Property Graphs. **[SIGMOD 2021]**[a]

Thank you!

[a] Joint work with the Property Schema Group.

References

Angles, Renzo. “The Property Graph Database Model”. In: *AMW*. Vol. 2100. CEUR Workshop Proceedings. 2018.

Bonifati, Angela et al. “Schema Validation and Evolution for Graph Databases”. In: *Conceptual Modeling - 38th International Conference, ER 2019*. Springer, 2019, pp. 448–456.

Dempster, Arthur and et al. “Maximum Likelihood from Incomplete Data via the EM Algorithm”. In: *Royal Statistical Society J.* 39 (1977), pp. 1–22.

Lbath, Hanâ, Angela Bonifati, and Russ Harmer. “Schema Inference for Property Graphs”. In: *EDBT*. 2021, pp. 499–504.