ENSSAT
LANNION

Learning Commonalities in RDF

Sara El Hassad François Goasdoué Hélène Jaudoin

IRISA, Univ. Rennes 1, Lannion, France

ESWC 2017 28th May - 1st June 2017

Introduction

Least general generalization (lgg)

- Machine Learning in the early 70's by Gordon Plotkin
- Knowledge representation domain in the early 90's
- Recently in semantic web

Introduction

Least general generalization (lgg)

- Machine Learning in the early 70's by Gordon Plotkin
- Knowledge representation domain in the early 90's
- Recently in semantic web

Applications of lgg

- Social context : lgg of users descriptions (profiles)
- Research common graph patterns between of datasets
- Linked Data Cloud : links between datasets

Introduction

Least general generalization (lgg)

- Machine Learning in the early 70's by Gordon Plotkin
- Knowledge representation domain in the early 90's
- Recently in semantic web

Applications of lgg

- Social context : lgg of users descriptions (profiles)
- Research common graph patterns between of datasets
- Linked Data Cloud : links between datasets

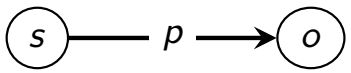
Goal

To study the problem in the setting of the *entire* RDF standard

Outline

RDF graphs

- Specification of RDF graphs with triples :
  $(\mathrm{s}, \mathrm{p}, \mathrm{o}) \in (\mathcal{U} \cup \mathcal{B}) \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{L} \cup \mathcal{B})$

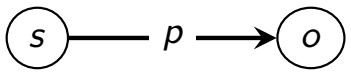


- Built-in property URIs to state RDF statements

RDF statement	Triple
Class assertion	$(\mathrm{s}, \mathrm{rdf{:}type}, \mathrm{o})$
Property assertion	$(\mathrm{s}, \mathrm{p}, \mathrm{o})$ with $\mathrm{p} \neq \mathrm{rdf{:}type}$

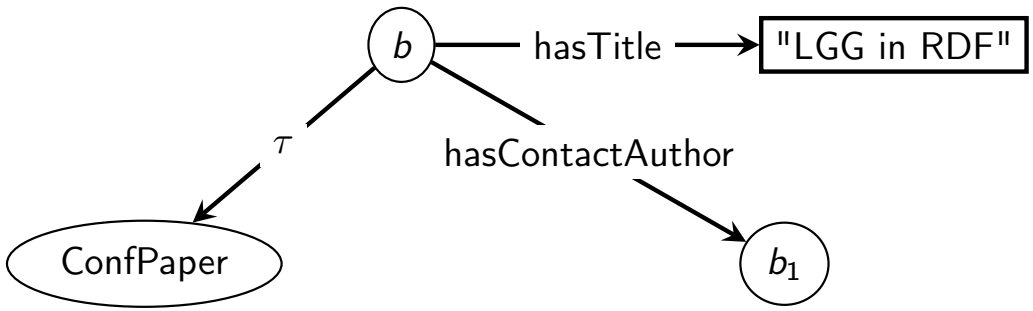
RDF graphs

- Specification of RDF graphs with triples :
  $(\mathrm{s}, \mathrm{p}, \mathrm{o}) \in (\mathcal{U} \cup \mathcal{B}) \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{L} \cup \mathcal{B})$



- Built-in property URIs to state RDF statements

RDF statement	Triple
Class assertion	$(\mathrm{s}, \mathrm{rdf{:}type}, \mathrm{o})$
Property assertion	$(\mathrm{s}, \mathrm{p}, \mathrm{o})$ with $\mathrm{p} \neq \mathrm{rdf{:}type}$



Adding ontological knowledge to RDF graphs

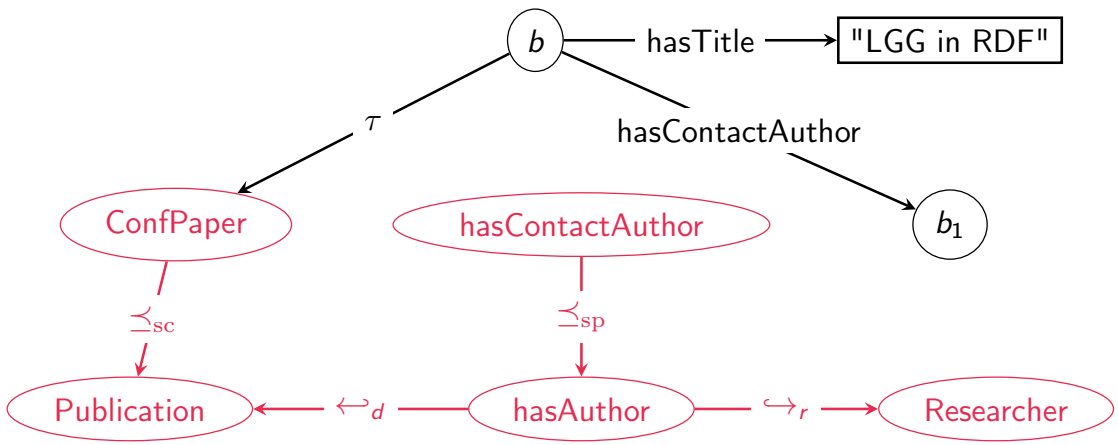
- Built-in property URIs to state RDF Schema statements, i.e., ontological constraints.

RDFS statement	Triple
Subclass	$(\mathsf{s}, \preceq_{\mathrm{sc}}, \mathsf{o})$
Subproperty	$(\mathsf{s}, \preceq_{\mathrm{sp}}, \mathsf{o})$
Domain typing	$(\mathsf{s}, \hookleftarrow_{d}, \mathsf{o})$
Range typing	$(\mathsf{s}, \hookrightarrow_{r}, \mathsf{o})$

Adding ontological knowledge to RDF graphs

- Built-in property URIs to state RDF Schema statements, i.e., ontological constraints.

RDFS statement	Triple
Subclass	$(s, \preceq_{sc}, o)$
Subproperty	$(s, \preceq_{sp}, o)$
Domain typing	$(s, \hookleftarrow_d, o)$
Range typing	$(s, \hookrightarrow_r, o)$



Deriving the implicit triples

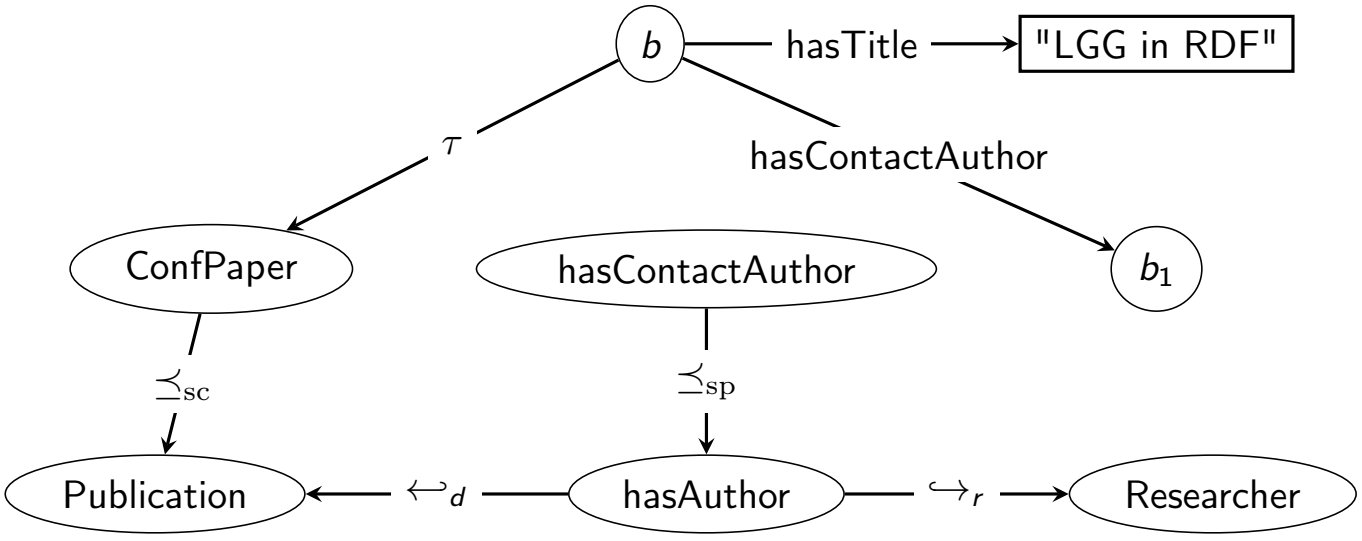


Figure: RDF graph $\mathcal{G}$

Deriving the implicit triples

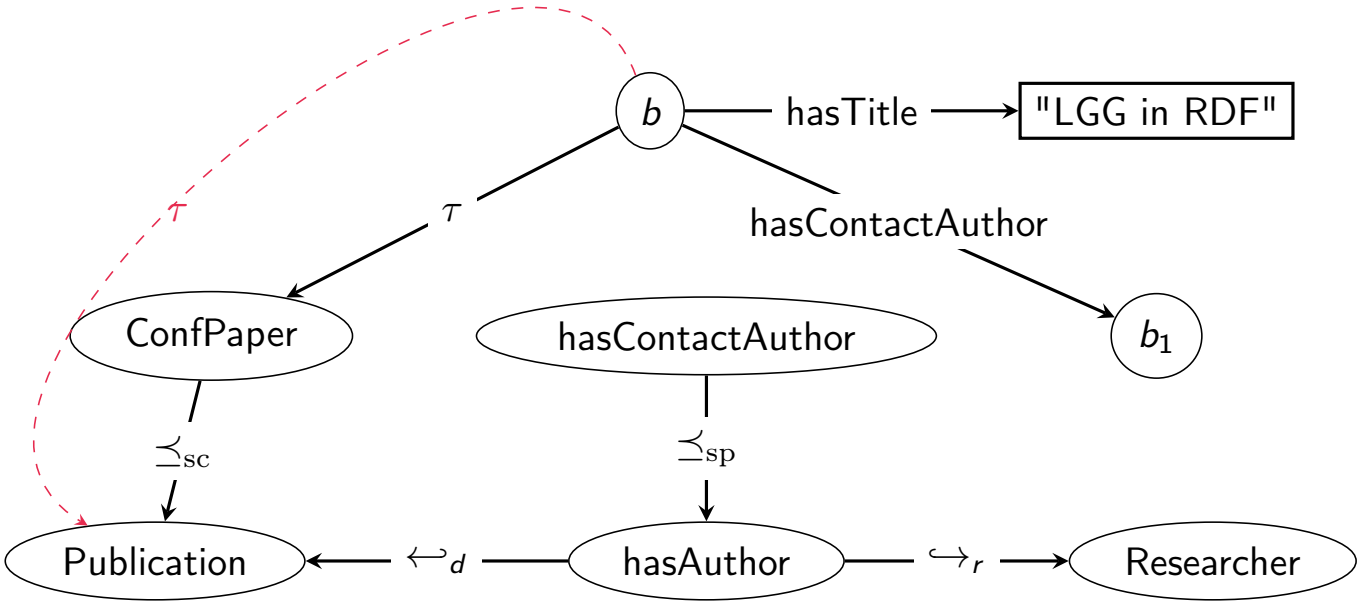


Figure: RDF graph $\mathcal{G}$

Deriving the implicit triples

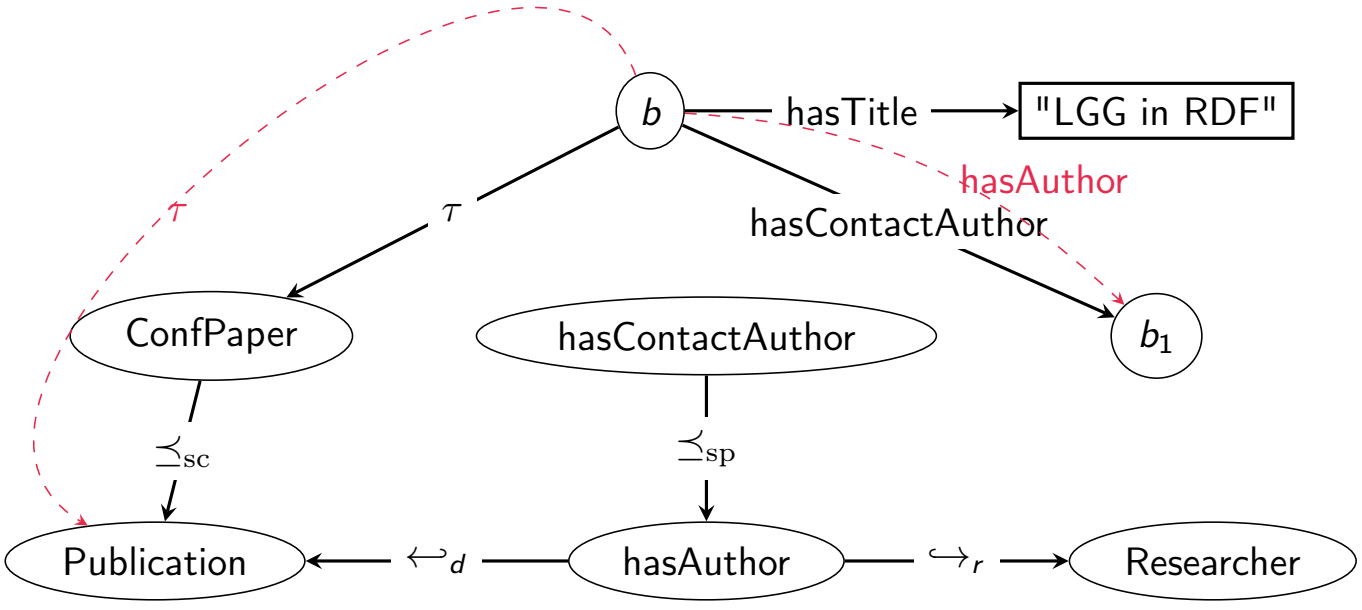


Figure: RDF graph $\mathcal{G}$

Deriving the implicit triples

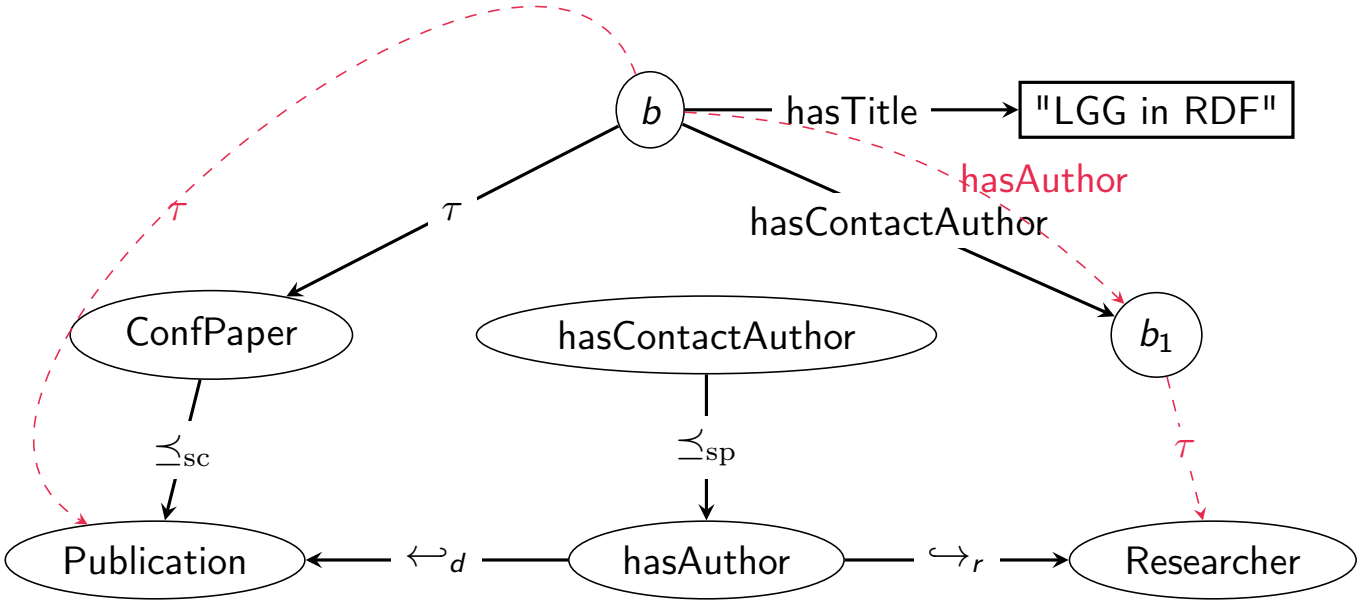


Figure: RDF graph $\mathcal{G}$

Deriving the implicit triples

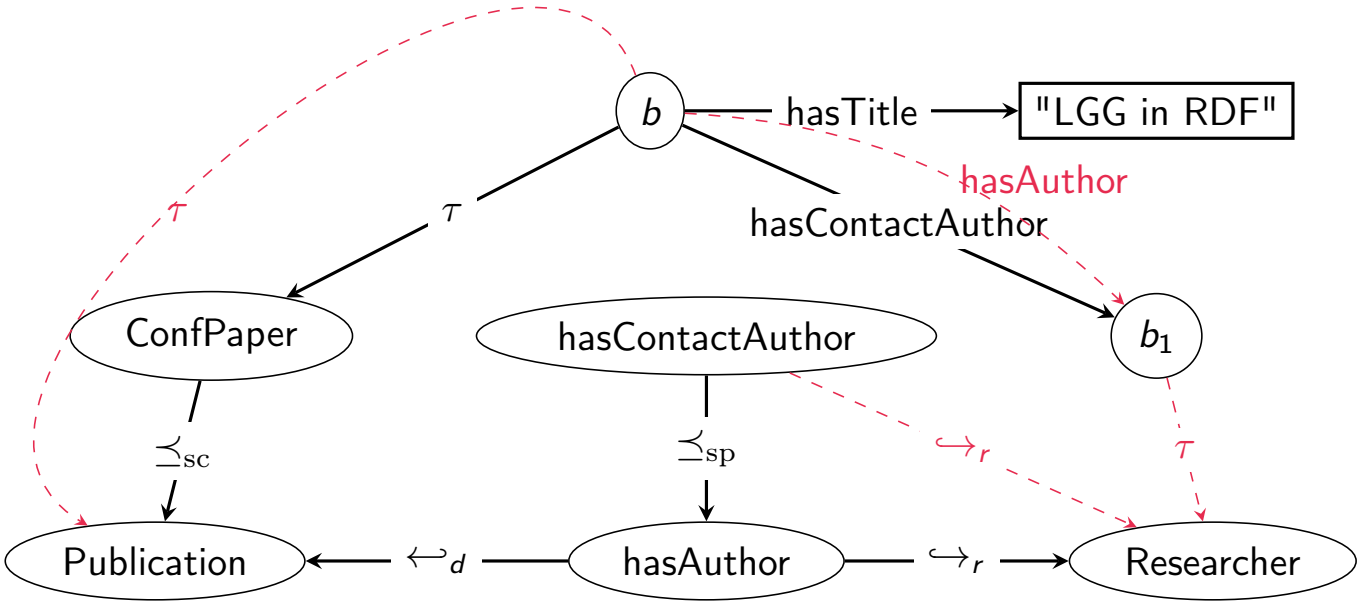


Figure: RDF graph $\mathcal{G}$

Deriving the implicit triples

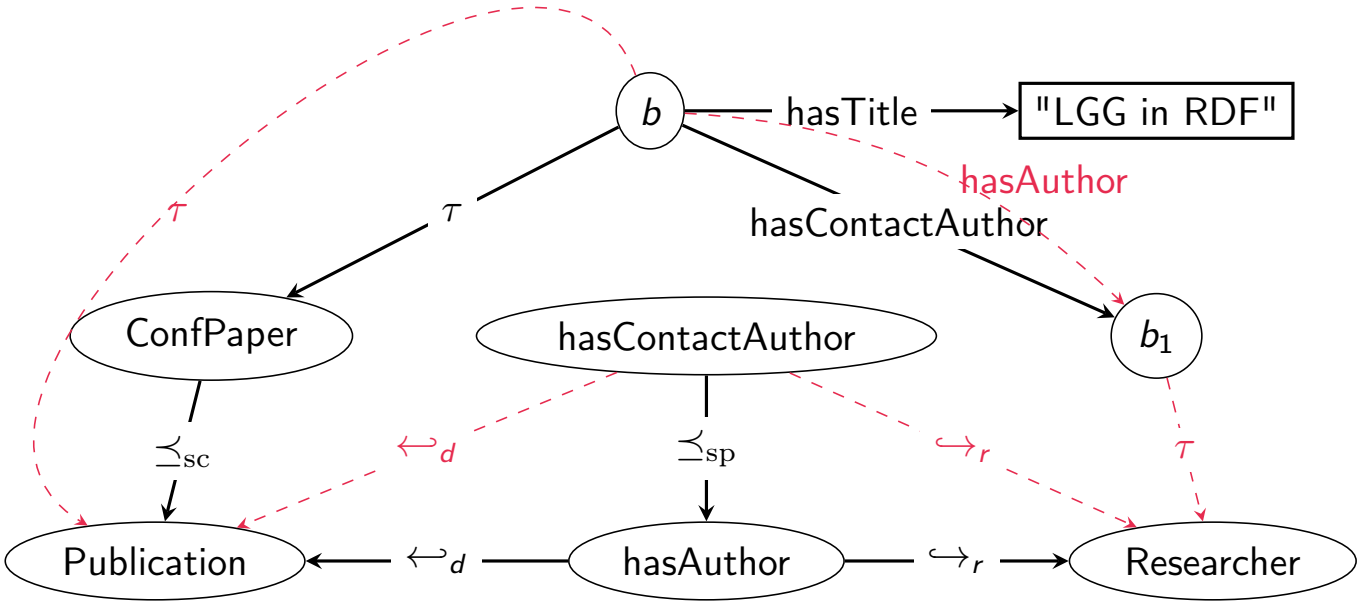


Figure: RDF graph $\mathcal{G}$

Deriving the implicit triples

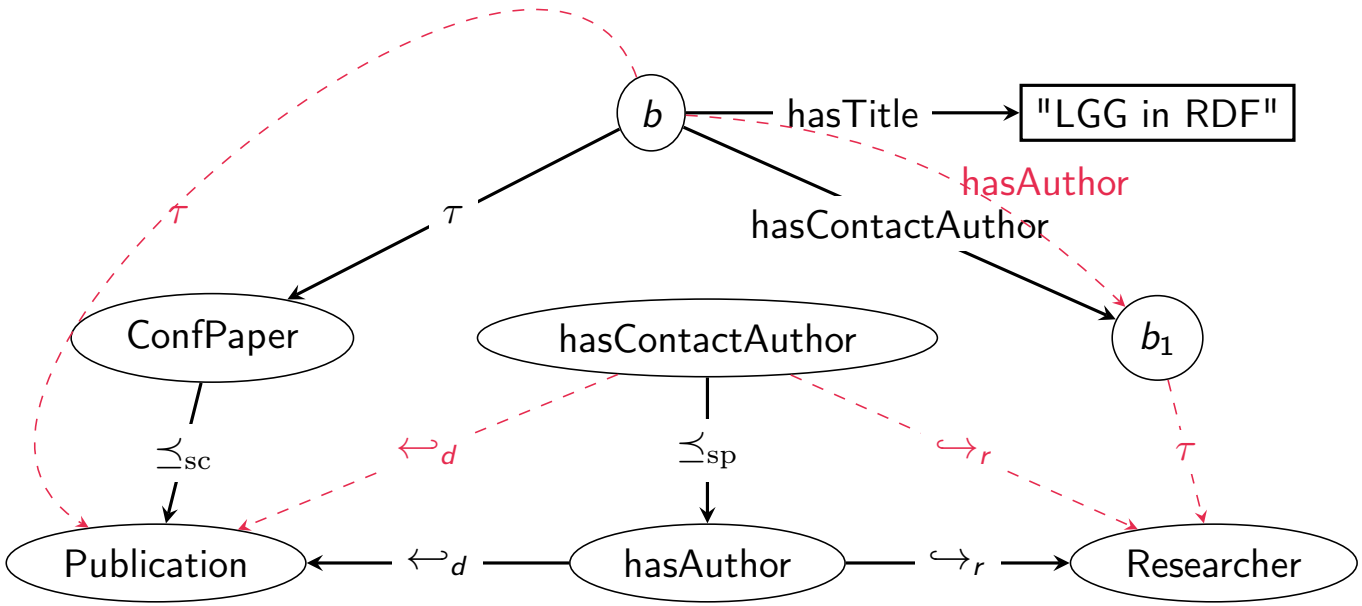


Figure: RDF graph $\mathcal{G}$

How to derive implicit triples of an RDF graph ?

Sample set of entailment rules

Rule [7]	Entailment rule
rdfs2	$(\mathtt{p}, \hookleftarrow_d, \mathtt{o}), (\mathtt{s}_1, \mathtt{p}, \mathtt{o}_1) \rightarrow (\mathtt{s}_1, \tau, \mathtt{o})$
rdfs3	$(\mathtt{p}, \hookrightarrow_r, \mathtt{o}), (\mathtt{s}_1, \mathtt{p}, \mathtt{o}_1) \rightarrow (\mathtt{o}_1, \tau, \mathtt{o})$
rdfs5	$(\mathtt{p}_1, \preceq_{\mathrm{sp}}, \mathtt{p}_2), (\mathtt{p}_2, \preceq_{\mathrm{sp}}, \mathtt{p}_3) \rightarrow (\mathtt{p}_1, \preceq_{\mathrm{sp}}, \mathtt{p}_3)$
rdfs7	$(\mathtt{p}_1, \preceq_{\mathrm{sp}}, \mathtt{p}_2), (\mathtt{s}, \mathtt{p}_1, \mathtt{o}) \rightarrow (\mathtt{s}, \mathtt{p}_2, \mathtt{o})$
rdfs9	$(\mathtt{s}, \preceq_{\mathrm{sc}}, \mathtt{o}), (\mathtt{s}_1, \tau, \mathtt{s}) \rightarrow (\mathtt{s}_1, \tau, \mathtt{o})$
rdfs11	$(\mathtt{s}, \preceq_{\mathrm{sc}}, \mathtt{o}), (\mathtt{o}, \preceq_{\mathrm{sc}}, \mathtt{o}_1) \rightarrow (\mathtt{s}, \preceq_{\mathrm{sc}}, \mathtt{o}_1)$
ext1	$(\mathtt{p}, \hookleftarrow_d, \mathtt{o}), (\mathtt{o}, \preceq_{\mathrm{sc}}, \mathtt{o}_1) \rightarrow (\mathtt{p}, \hookleftarrow_d, \mathtt{o}_1)$
ext2	$(\mathtt{p}, \hookrightarrow_r, \mathtt{o}), (\mathtt{o}, \preceq_{\mathrm{sc}}, \mathtt{o}_1) \rightarrow (\mathtt{p}, \hookrightarrow_r, \mathtt{o}_1)$
ext3	$(\mathtt{p}, \preceq_{\mathrm{sp}}, \mathtt{p}_1), (\mathtt{p}_1, \hookleftarrow_d, \mathtt{o}) \rightarrow (\mathtt{p}, \hookleftarrow_d, \mathtt{o})$
ext4	$(\mathtt{p}, \preceq_{\mathrm{sp}}, \mathtt{p}_1), (\mathtt{p}_1, \hookrightarrow_r, \mathtt{o}) \rightarrow (\mathtt{p}, \hookrightarrow_r, \mathtt{o})$

Semantics of RDF graphs

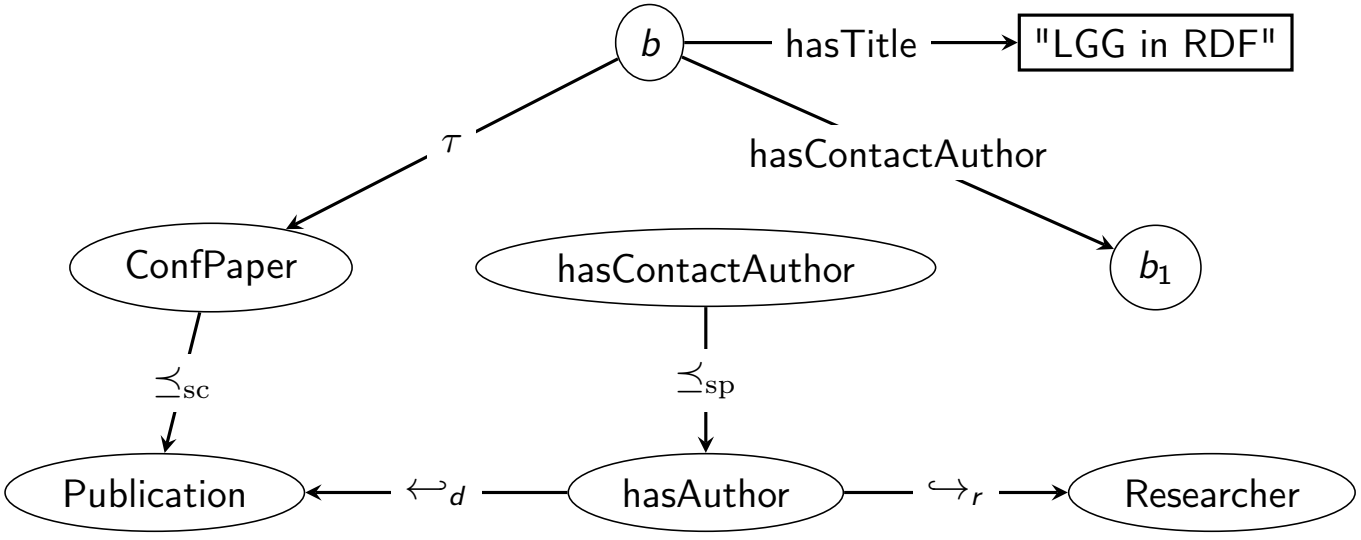


Figure: RDF graph $\mathcal{G}$

Semantics of RDF graphs

$$rdfs9 : (\mathtt{s}, \preceq_{\mathrm{sc}}, \mathtt{o}), (\mathtt{s}_1, \tau, \mathtt{s}) \rightarrow (\mathtt{s}_1, \tau, \mathtt{o})$$

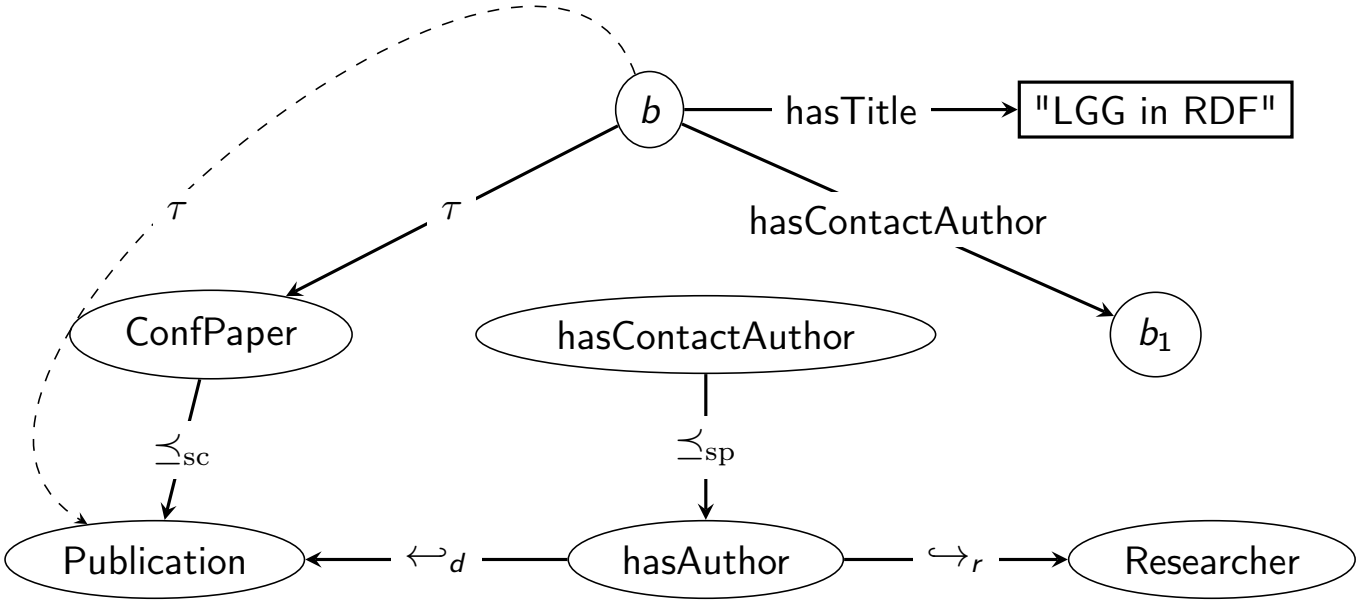


Figure: RDF graph $\mathcal{G}$

Semantics of RDF graphs

$$rdfs7 : (\mathrm{p}_1, \preceq_{\mathrm{sp}}, \mathrm{p}_2), (\mathrm{s}, \mathrm{p}_1, \mathrm{o}) \rightarrow (\mathrm{s}, \mathrm{p}_2, \mathrm{o})$$

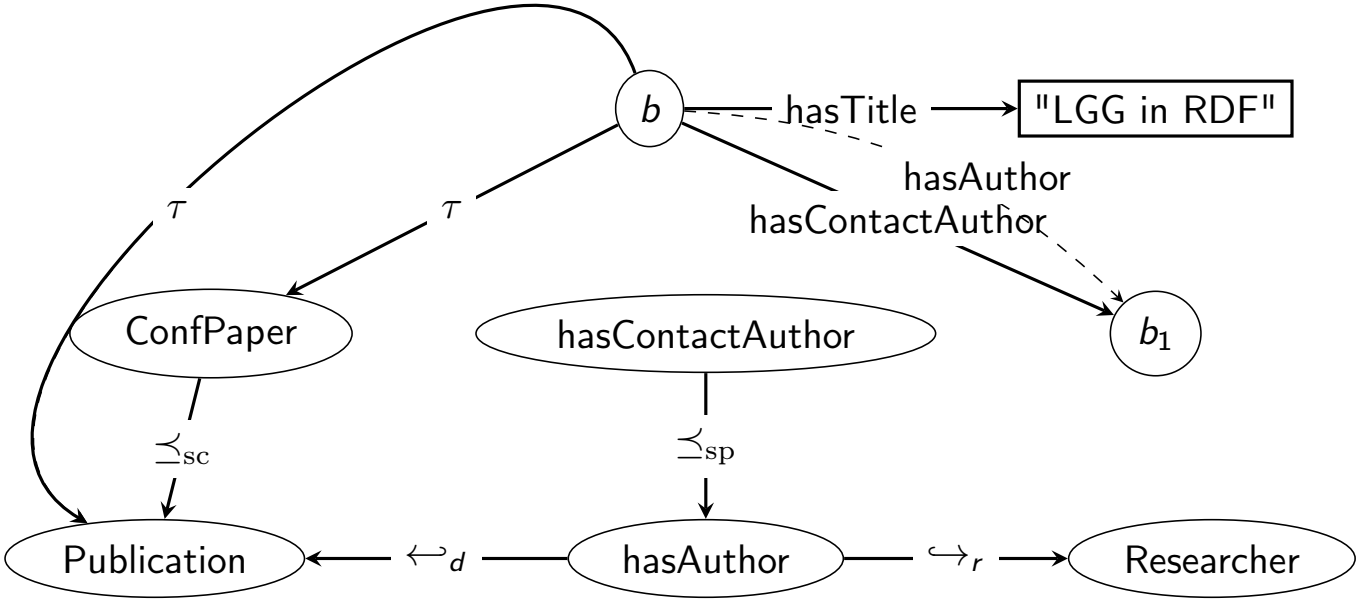


Figure: RDF graph $\mathcal{G}$

Semantics of RDF graphs

$$rdfs3 : (\mathrm{p}, \hookrightarrow_r, \mathrm{o}), (\mathrm{s}_1, \mathrm{p}, \mathrm{o}_1) \rightarrow (\mathrm{o}_1, \tau, \mathrm{o})$$

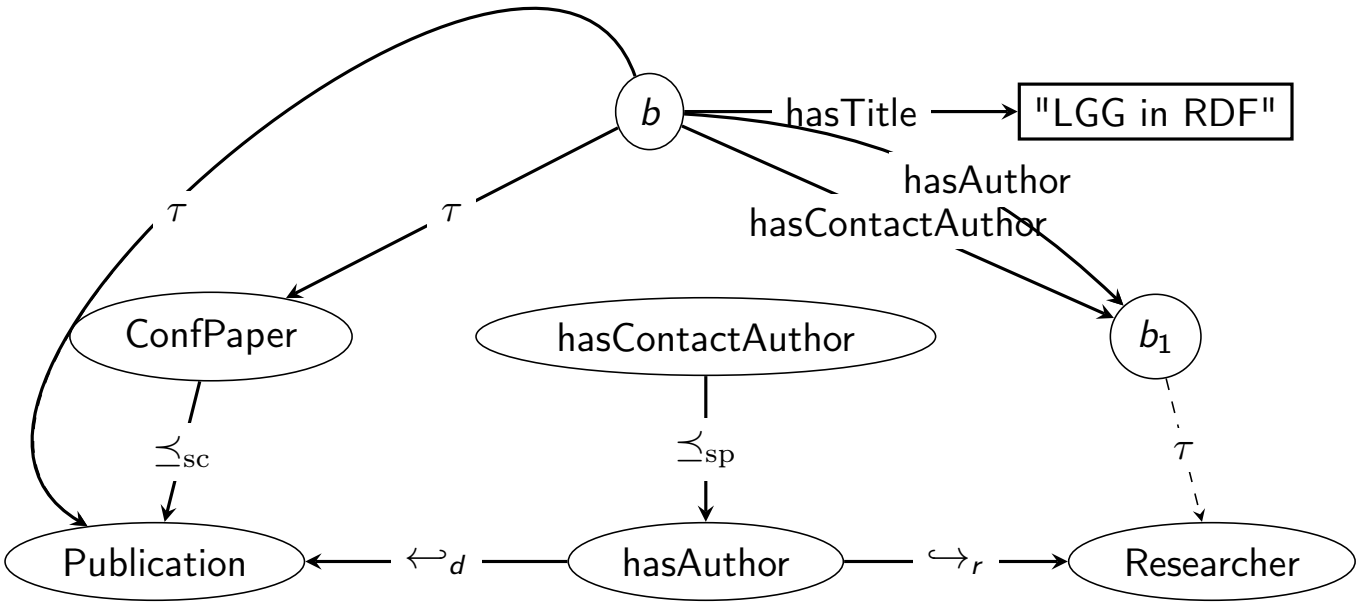


Figure: RDF graph $\mathcal{G}$

Semantics of RDF graphs

$$ext4 : (p, \preceq_{sp}, p_1), (p_1, \hookrightarrow_r, o) \rightarrow (p, \hookrightarrow_r, o)$$

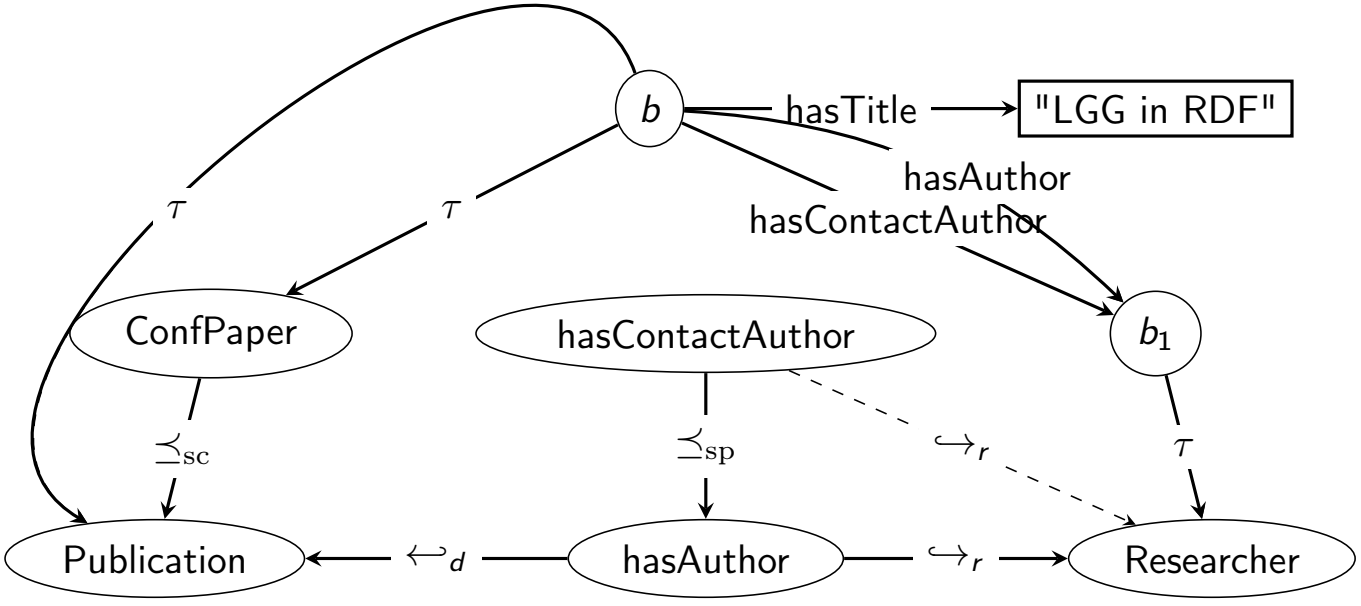


Figure: RDF graph $\mathcal{G}$

Semantics of RDF graphs

$$ext3 : (p, \preceq_{sp}, p_1), (p_1, \hookleftarrow_d, o) \rightarrow (p, \hookleftarrow_d, o)$$

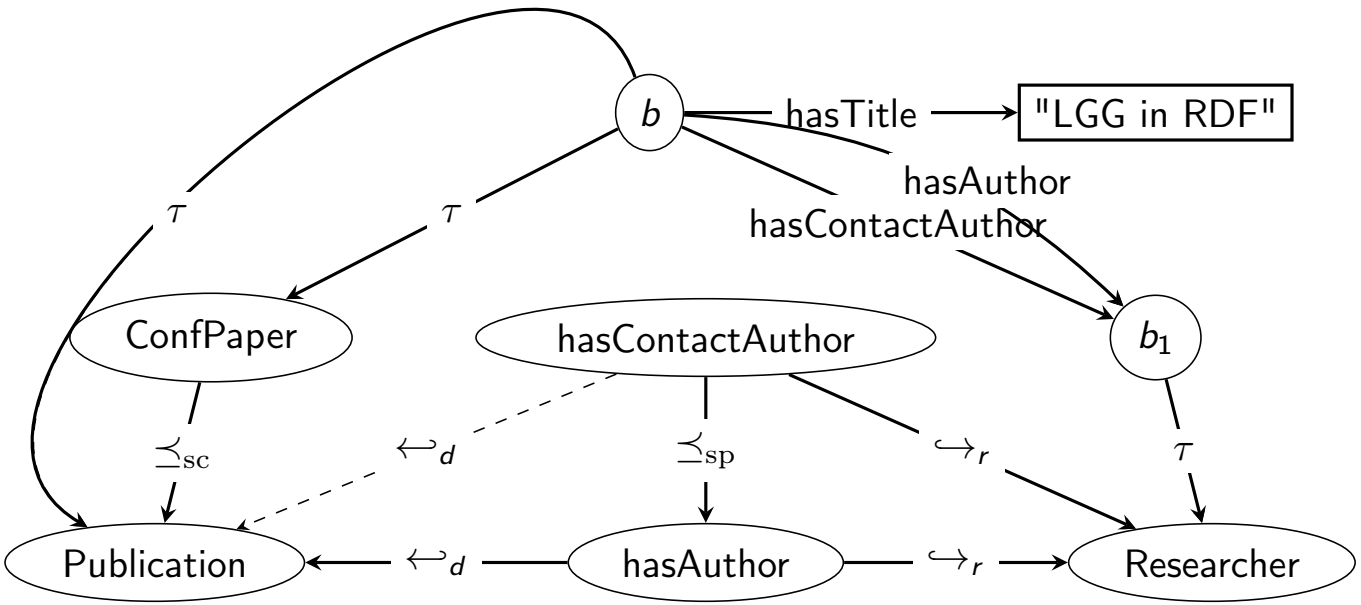


Figure: RDF graph $\mathcal{G}$

Semantics of RDF graphs

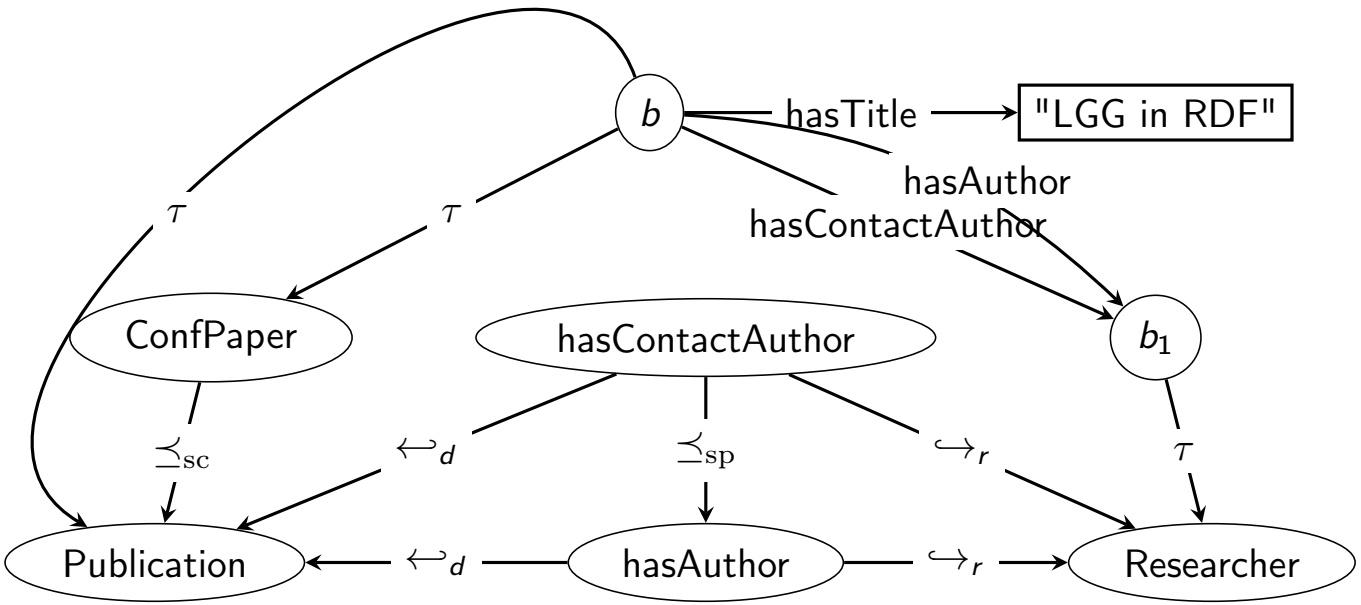


Figure: Saturated RDF graph $\mathcal{G}^{\infty}$

Entailment between RDF graphs

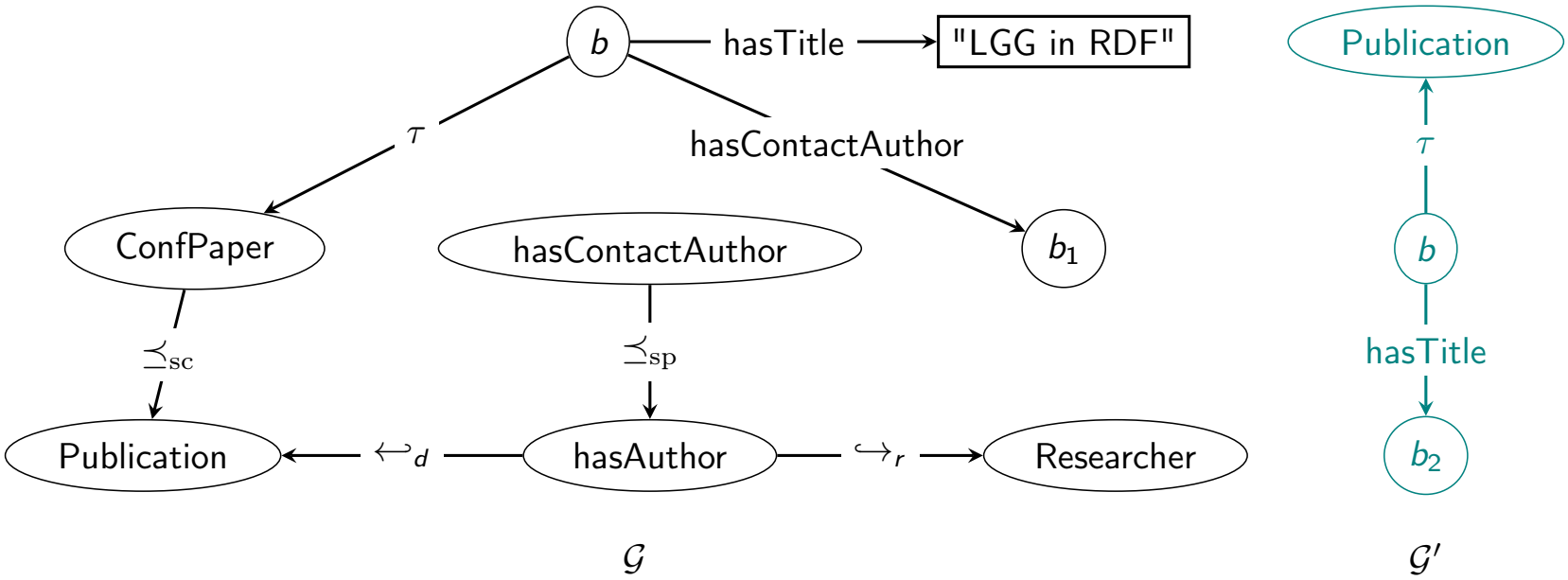
Let $\mathcal{G}$ and $\mathcal{G}'$ be two graphs RDF and $\mathcal{R}$ a set of RDF entailment rules. There exists relationship to compare $\mathcal{G}$ and $\mathcal{G}'$ called *entailment between graphs*
$\mathcal{G}$ is more specific than $\mathcal{G}'$:

- $\mathcal{G} \models_{\mathcal{R}} \mathcal{G}' \iff \mathcal{G}^{\infty} \models \mathcal{G}'$

There must exist an embedding of $\mathcal{G}'$ in $\mathcal{G}^{\infty}$.

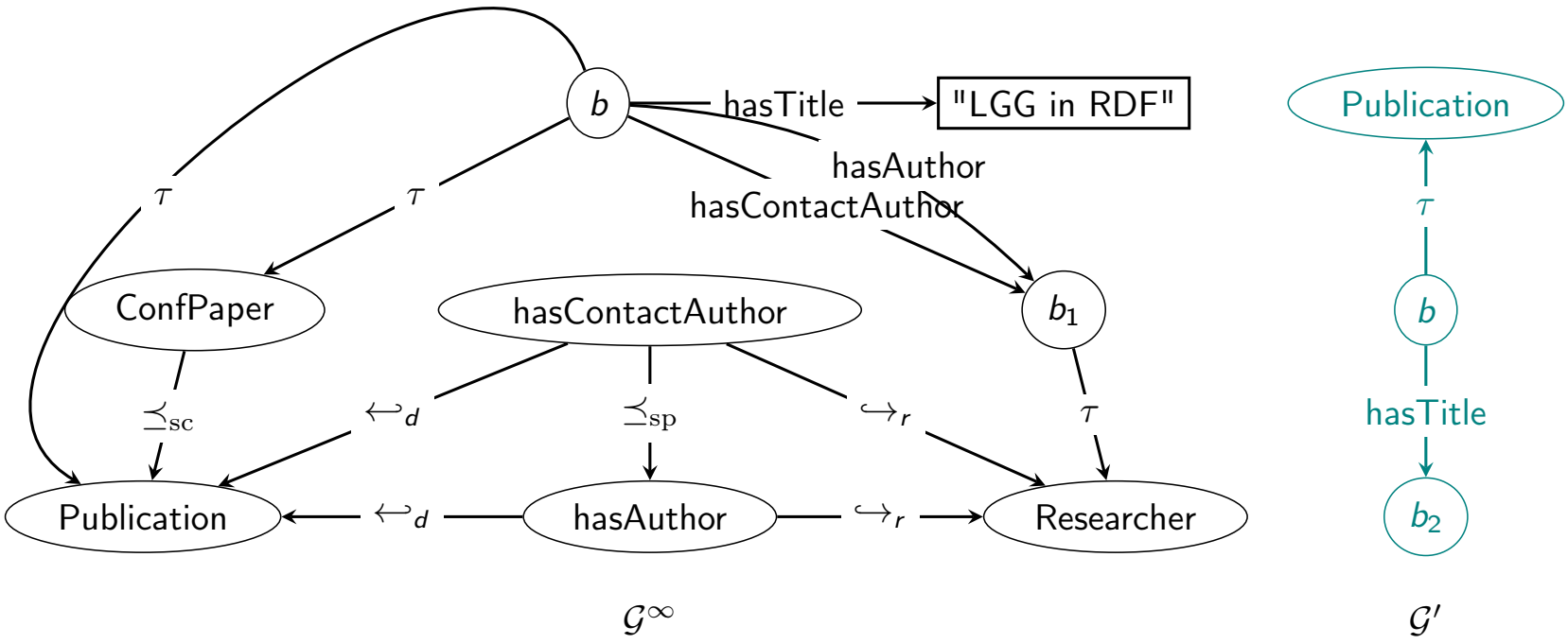
Entailment between RDF graphs

$$\mathcal{G} \overset{?}{\models}_{\mathcal{R}} \mathcal{G}'$$



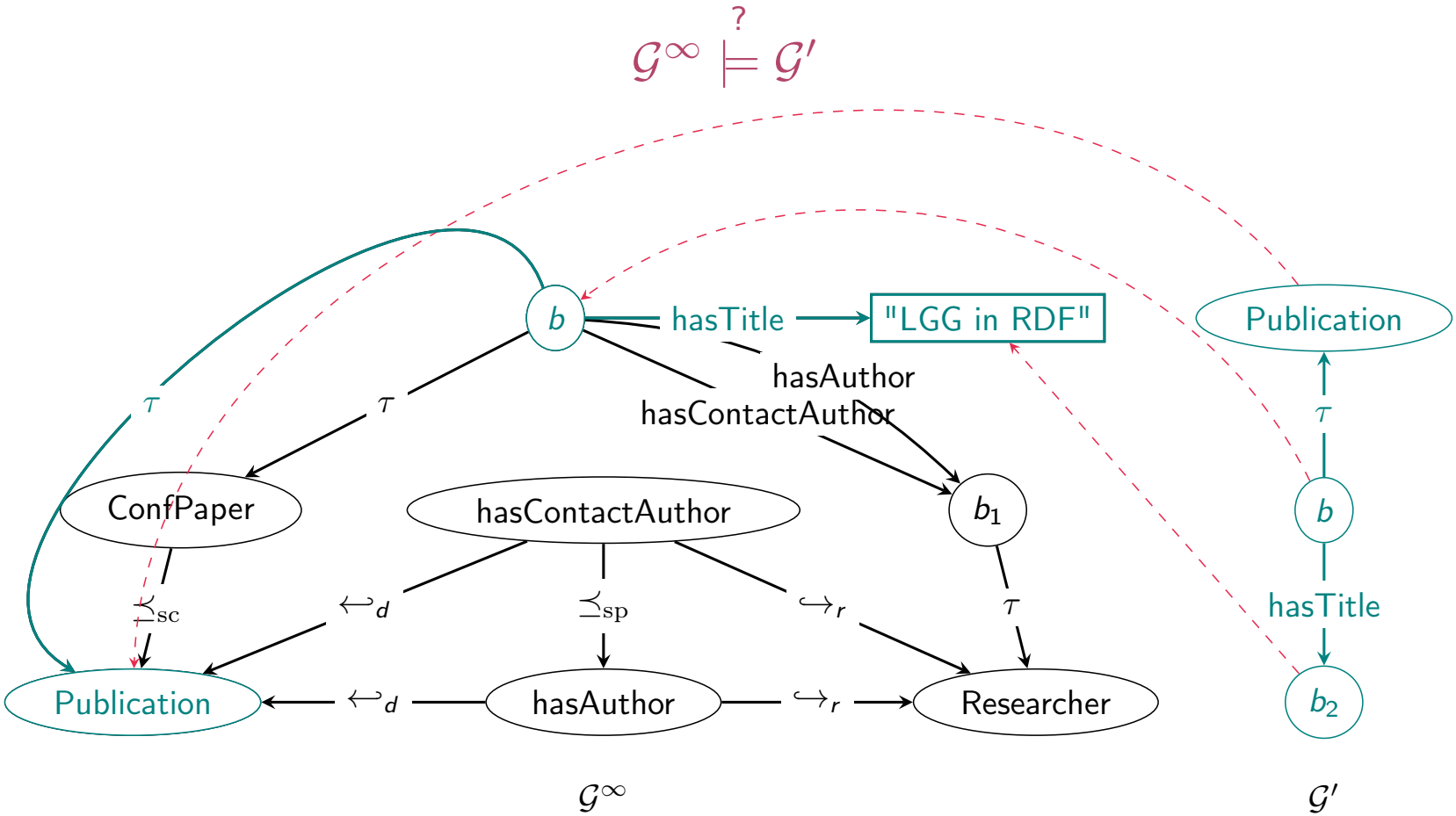
Entailment between RDF graphs

$$\mathcal{G}^{\infty} \overset{?}{\models} \mathcal{G}'$$



$\mathcal{G}^{\infty}$ $\mathcal{G}'$

Entailment between RDF graphs



RDF graph $\mathcal{G}$ is more specific than RDF graph $\mathcal{G}'$

Outline

Towards defining lgg in RDF

A *least general generalization* (lgg) of n descriptions $d_1, \ldots, d_n$ is a most specific description d generalizing every $d_{1 \leq i \leq n}$ for some generalization/specialization relation between descriptions (G.Plotkin).

lgg in RDF

- descriptions are RDF graphs
- relation generalization/specialization is entailment between RDF graphs

Defining the lgg of RDF graphs

Definition (lgg of RDF graphs)

Let $\mathcal{G}_1, \ldots, \mathcal{G}_n$ be RDF graphs and $\mathcal{R}$ a set of RDF entailment rules.

- A *generalization* of $\mathcal{G}_1, \ldots, \mathcal{G}_n$ is an RDF graph $\mathcal{G}_g$ such that $\mathcal{G}_i \models_{\mathcal{R}} \mathcal{G}_g$ holds for $1 \leq i \leq n$.
- A *least general generalization* (lgg) of $\mathcal{G}_1, \ldots, \mathcal{G}_n$ is a generalization $\mathcal{G}_{\text{lgg}}$ of $\mathcal{G}_1, \ldots, \mathcal{G}_n$ such that for any other generalization $\mathcal{G}_g$ of $\mathcal{G}_1, \ldots, \mathcal{G}_n$, $\mathcal{G}_{\text{lgg}} \models_{\mathcal{R}} \mathcal{G}_g$ holds.

Result : lgg of n RDF graphs vs lgg of two RDF graphs

$$
\begin{array}{ll}
\ell_3(\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3) & \equiv_{\mathcal{R}} \ \ell_2(\ell_2(\mathcal{G}_1, \mathcal{G}_2), \mathcal{G}_3) \\
\ldots & \ldots \\
\ell_n(\mathcal{G}_1, \ldots, \mathcal{G}_n) & \equiv_{\mathcal{R}} \ \ell_2(\ell_{n-1}(\mathcal{G}_1, \ldots, \mathcal{G}_{n-1}), \mathcal{G}_n) \\
& \equiv_{\mathcal{R}} \ \ell_2(\ell_2(\cdots \ell_2(\ell_2(\mathcal{G}_1, \mathcal{G}_2), \mathcal{G}_3) \cdots, \mathcal{G}_{n-1}), \mathcal{G}_n)
\end{array}
$$

Defining the lgg of RDF graphs

Definition (lgg of RDF graphs)

Let $\mathcal{G}_1, \ldots, \mathcal{G}_n$ be RDF graphs and $\mathcal{R}$ a set of RDF entailment rules.

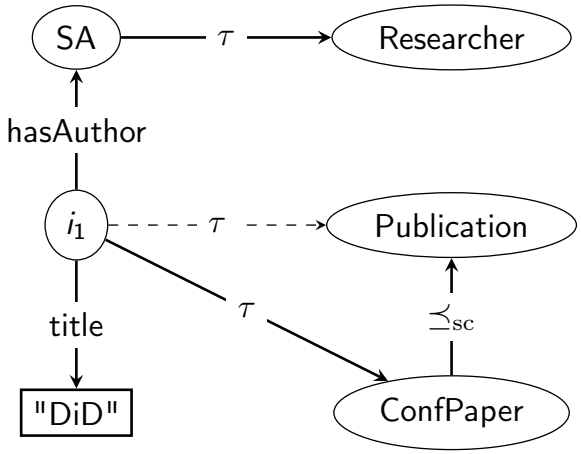
- A *generalization* of $\mathcal{G}_1, \ldots, \mathcal{G}_n$ is an RDF graph $\mathcal{G}_g$ such that $\mathcal{G}_i \models_{\mathcal{R}} \mathcal{G}_g$ holds for $1 \leq i \leq n$.
- A *least general generalization* (lgg) of $\mathcal{G}_1, \ldots, \mathcal{G}_n$ is a generalization $\mathcal{G}_{\text{lgg}}$ of $\mathcal{G}_1, \ldots, \mathcal{G}_n$ such that for any other generalization $\mathcal{G}_g$ of $\mathcal{G}_1, \ldots, \mathcal{G}_n$, $\mathcal{G}_{\text{lgg}} \models_{\mathcal{R}} \mathcal{G}_g$ holds.

Result : lgg of n RDF graphs vs lgg of two RDF graphs

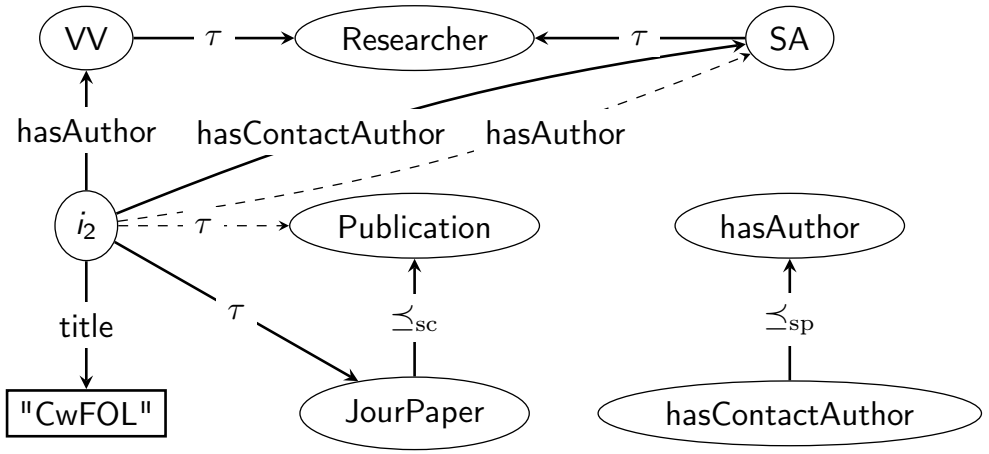
$$
\begin{array}{lll}
\ell_3(\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3) & \equiv_{\mathcal{R}} & \ell_2(\ell_2(\mathcal{G}_1, \mathcal{G}_2), \mathcal{G}_3) \\
\ldots & \ldots & \\
\ell_n(\mathcal{G}_1, \ldots, \mathcal{G}_n) & \equiv_{\mathcal{R}} & \ell_2(\ell_{n-1}(\mathcal{G}_1, \ldots, \mathcal{G}_{n-1}), \mathcal{G}_n) \\
 & \equiv_{\mathcal{R}} & \ell_2(\ell_2(\cdots \ell_2(\ell_2(\mathcal{G}_1, \mathcal{G}_2), \mathcal{G}_3) \cdots, \mathcal{G}_{n-1}), \mathcal{G}_n)
\end{array}
$$

We focus on computing lgg of two RDF graphs

Defining the lgg of RDF graphs

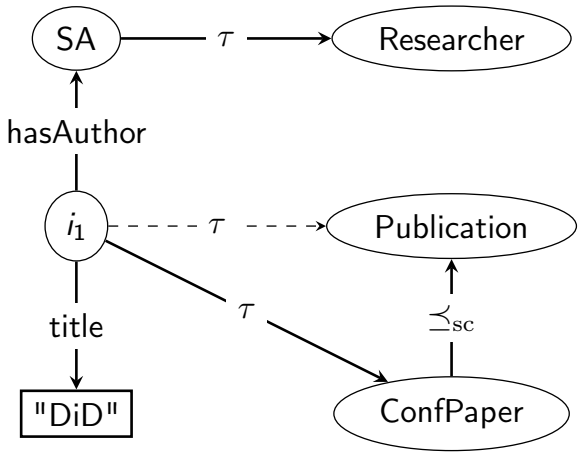


$\mathcal{G}_1$

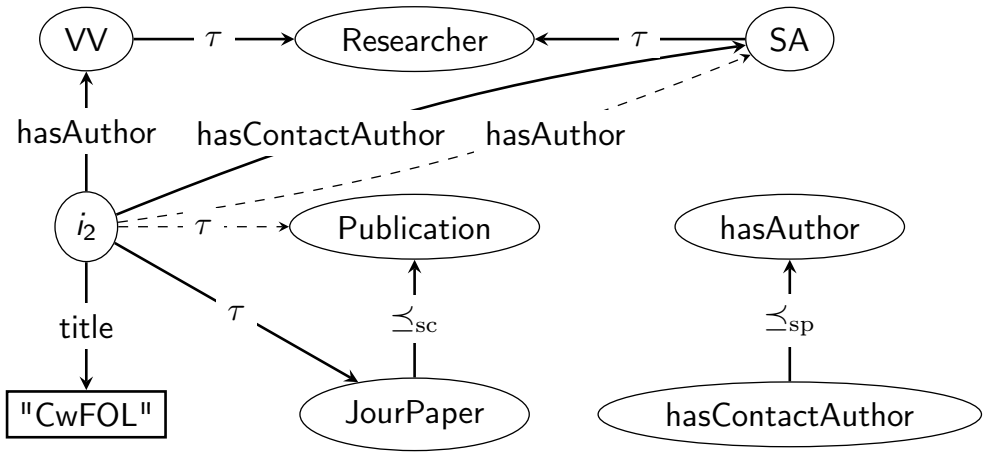


$\mathcal{G}_2$

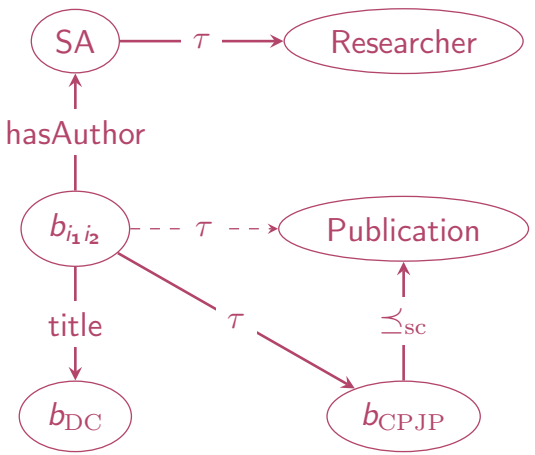
Defining the lgg of RDF graphs



$\mathcal{G}_1$

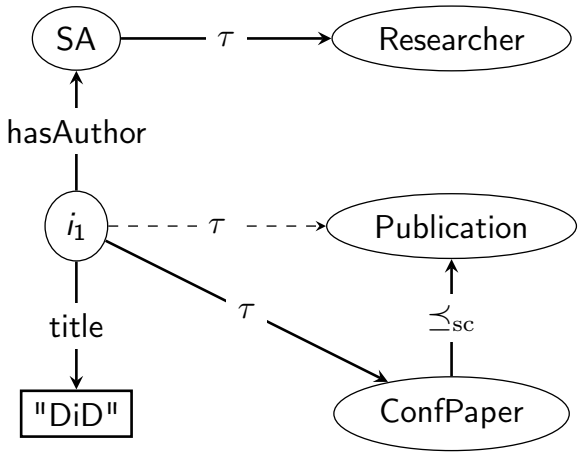


$\mathcal{G}_2$

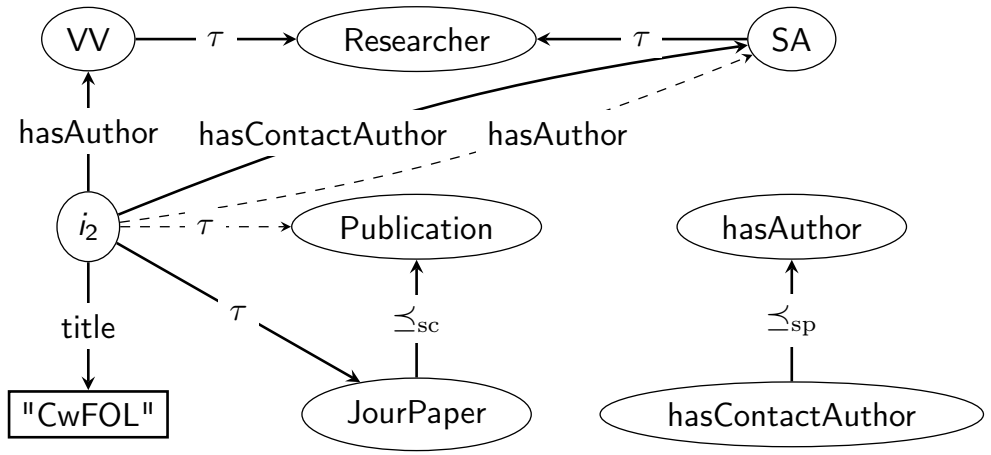


$\mathcal{G}_{\mathtt{lgg}}$

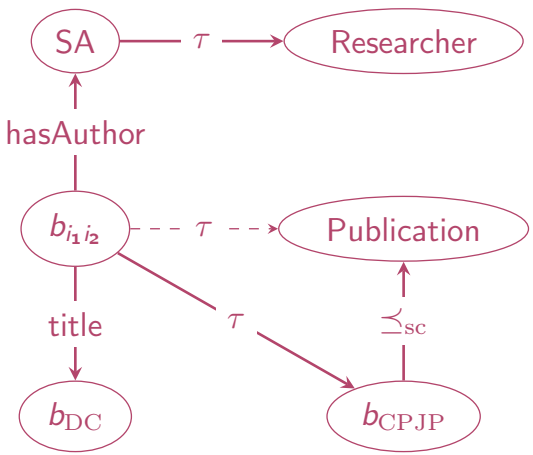
Defining the lgg of RDF graphs



$\mathcal{G}_1$



$\mathcal{G}_2$



$\mathcal{G}_{\mathtt{lgg}}$

How to compute this graph ?

The cover graph of RDF graphs

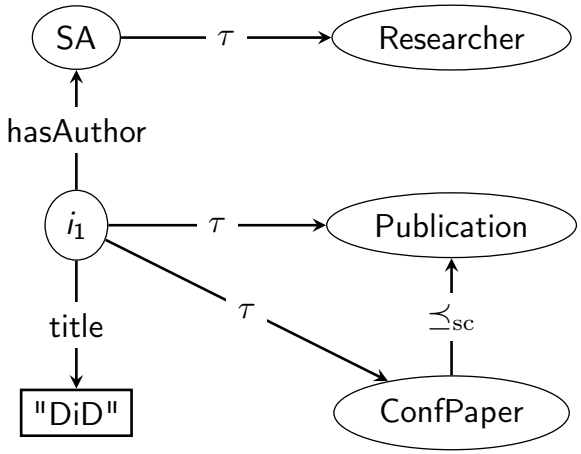
Definition (Cover graph)

The *cover graph* $\mathcal{G}$ of two RDF graph $\mathcal{G}_1$ and $\mathcal{G}_2$ is the RDF graph such that for every property p in both $\mathcal{G}_1$ and $\mathcal{G}_2$:

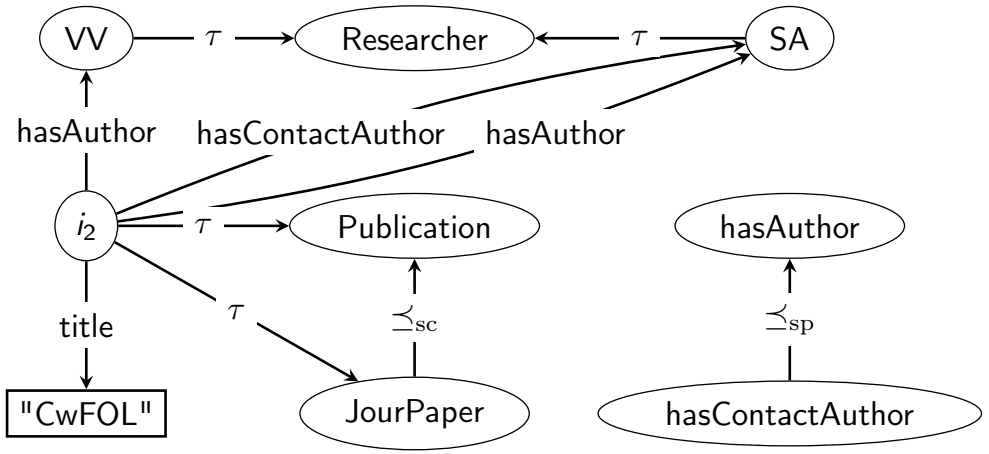
$$(t_1, p, t_2) \in \mathcal{G}_1 \text{ and } (t_3, p, t_4) \in \mathcal{G}_2 \text{ iff } (t_5, p, t_6) \in \mathcal{G}$$

with $t_5 = t_1$ if $t_1 = t_3$ and $t_1 \in \mathcal{U} \cup \mathcal{L}$, else t_5 is the blank node $b_{t_1 t_3}$, and, similarly $t_6 = t_2$ if $t_2 = t_4$ and $t_2 \in \mathcal{U} \cup \mathcal{L}$, else t_6 is the blank node $b_{t_2 t_4}$.

The cover graph of RDF graphs

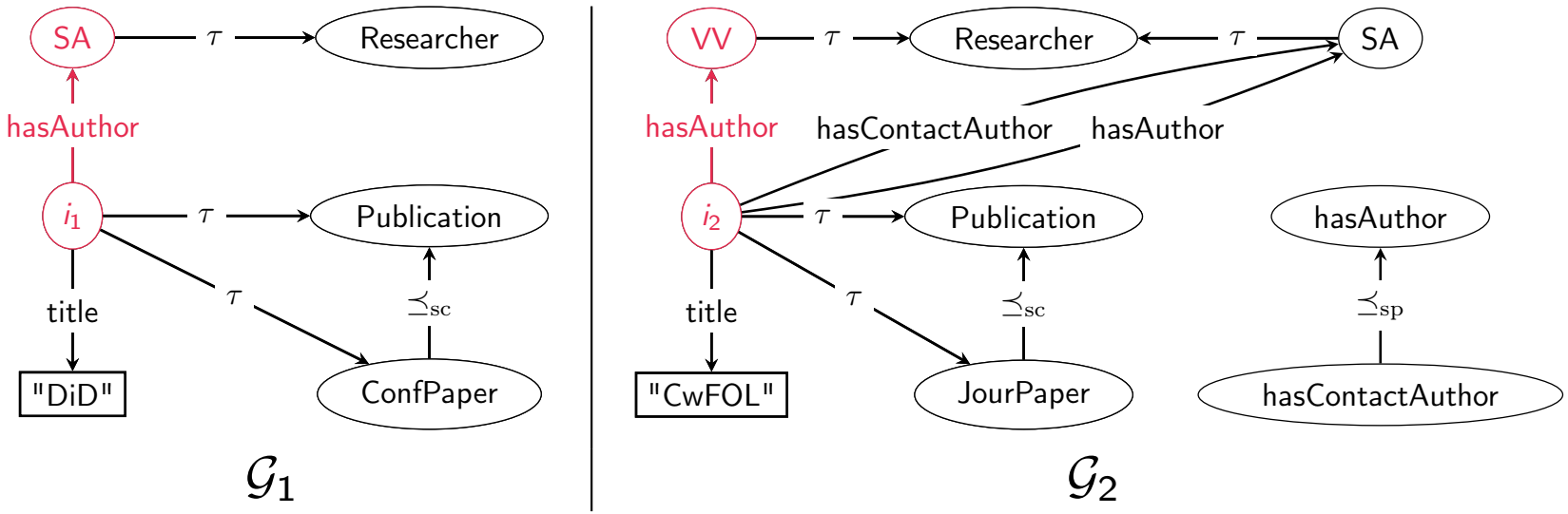


$\mathcal{G}_1$

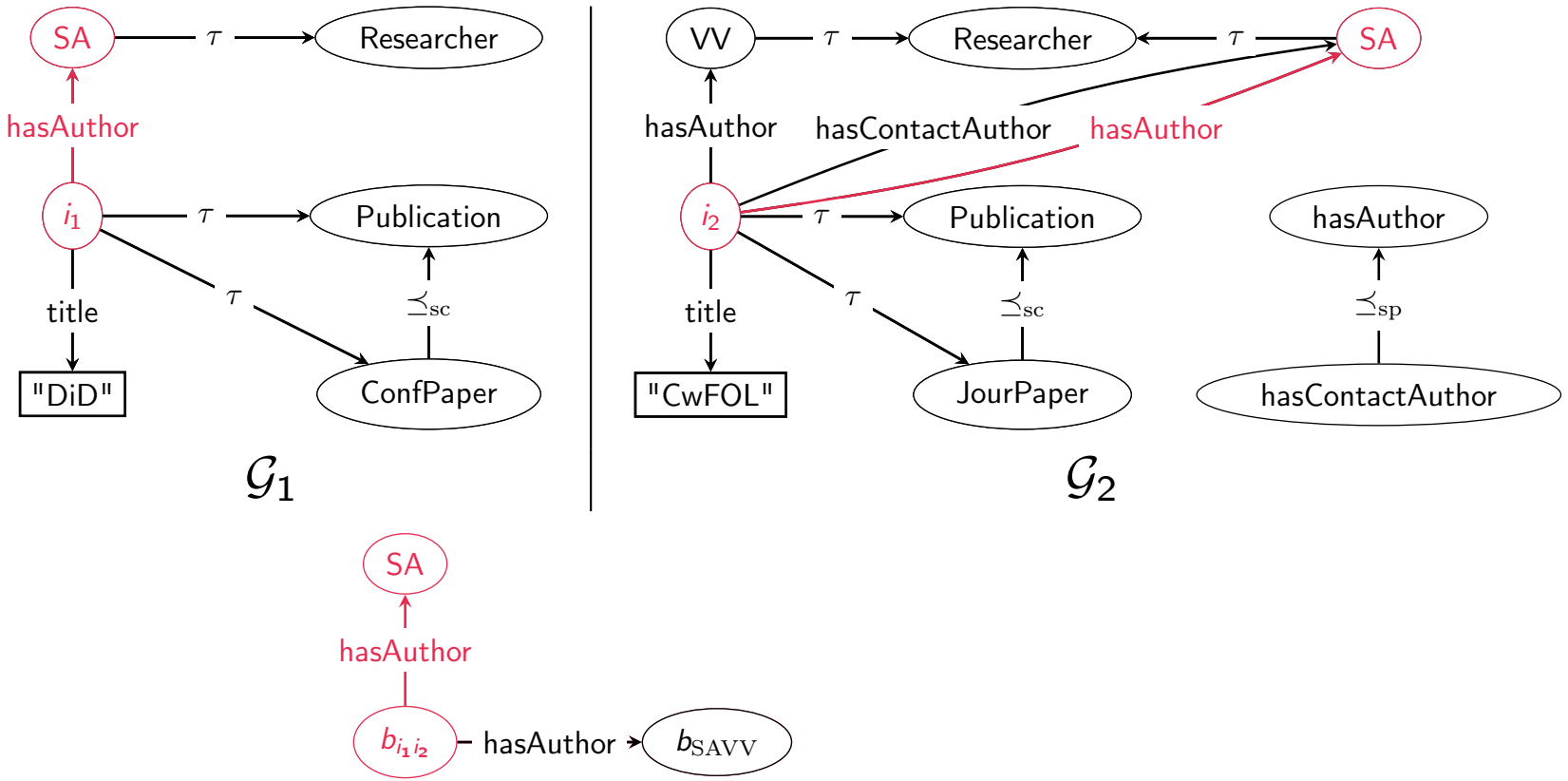


$\mathcal{G}_2$

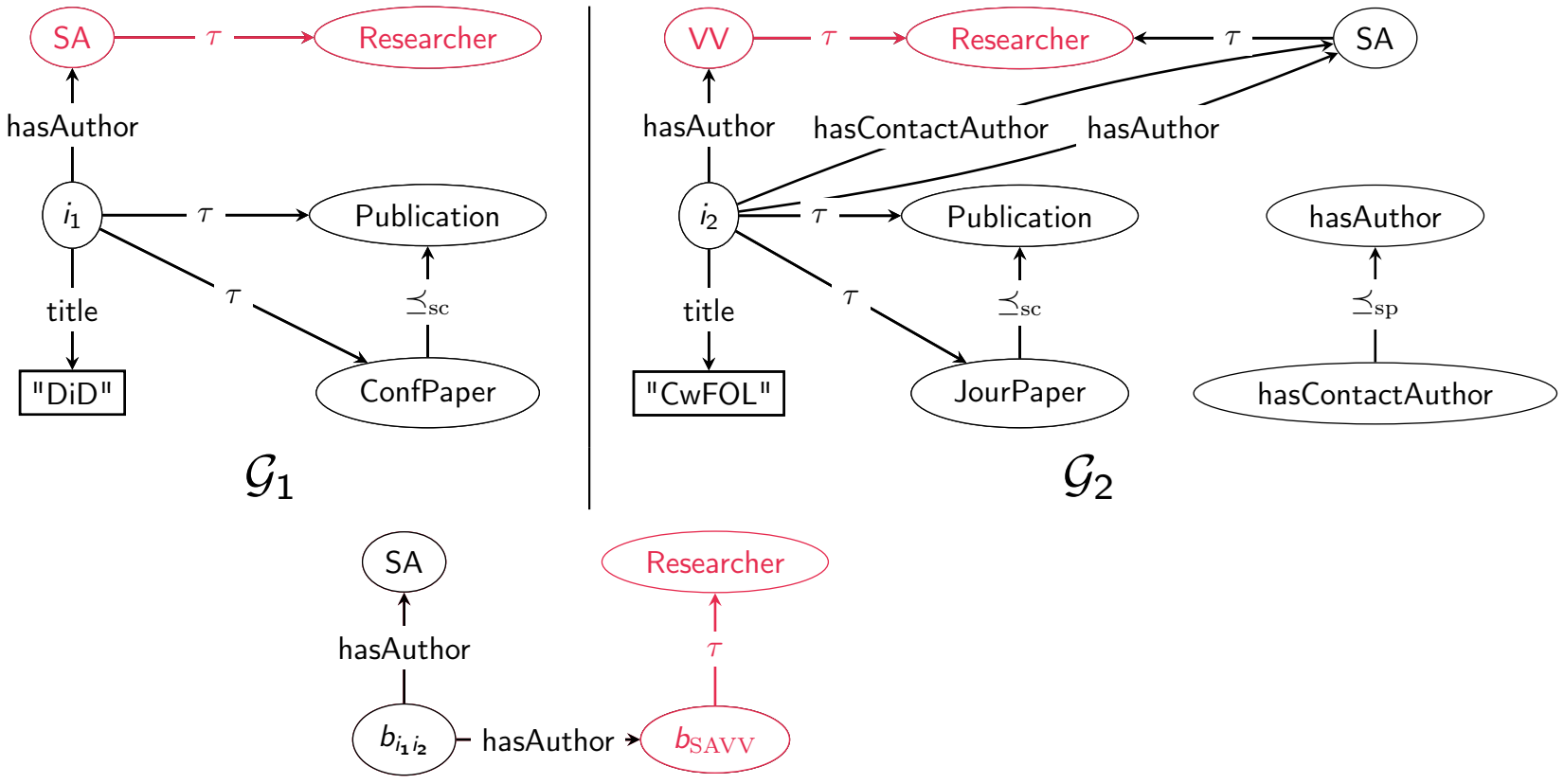
The cover graph of RDF graphs



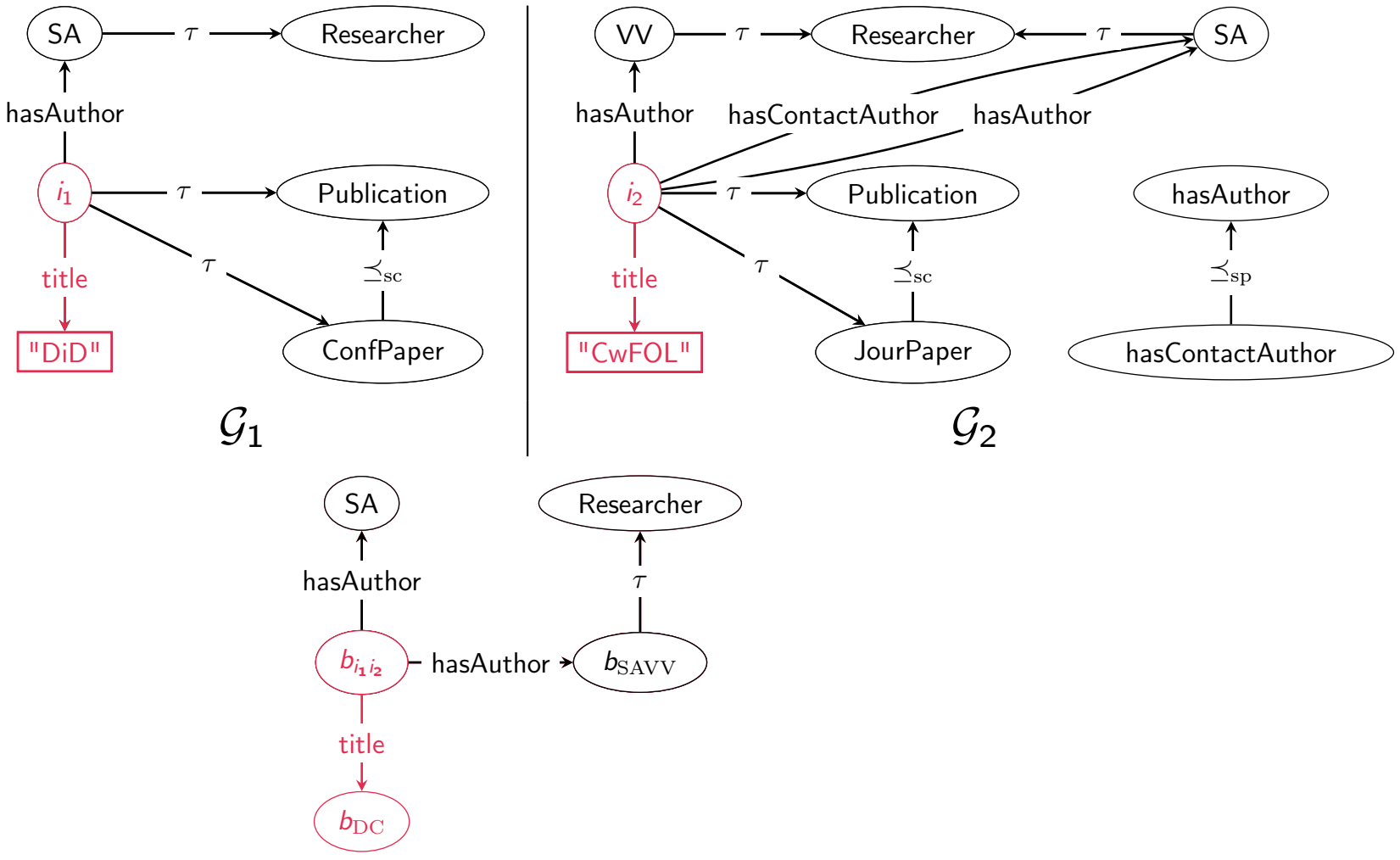
The cover graph of RDF graphs



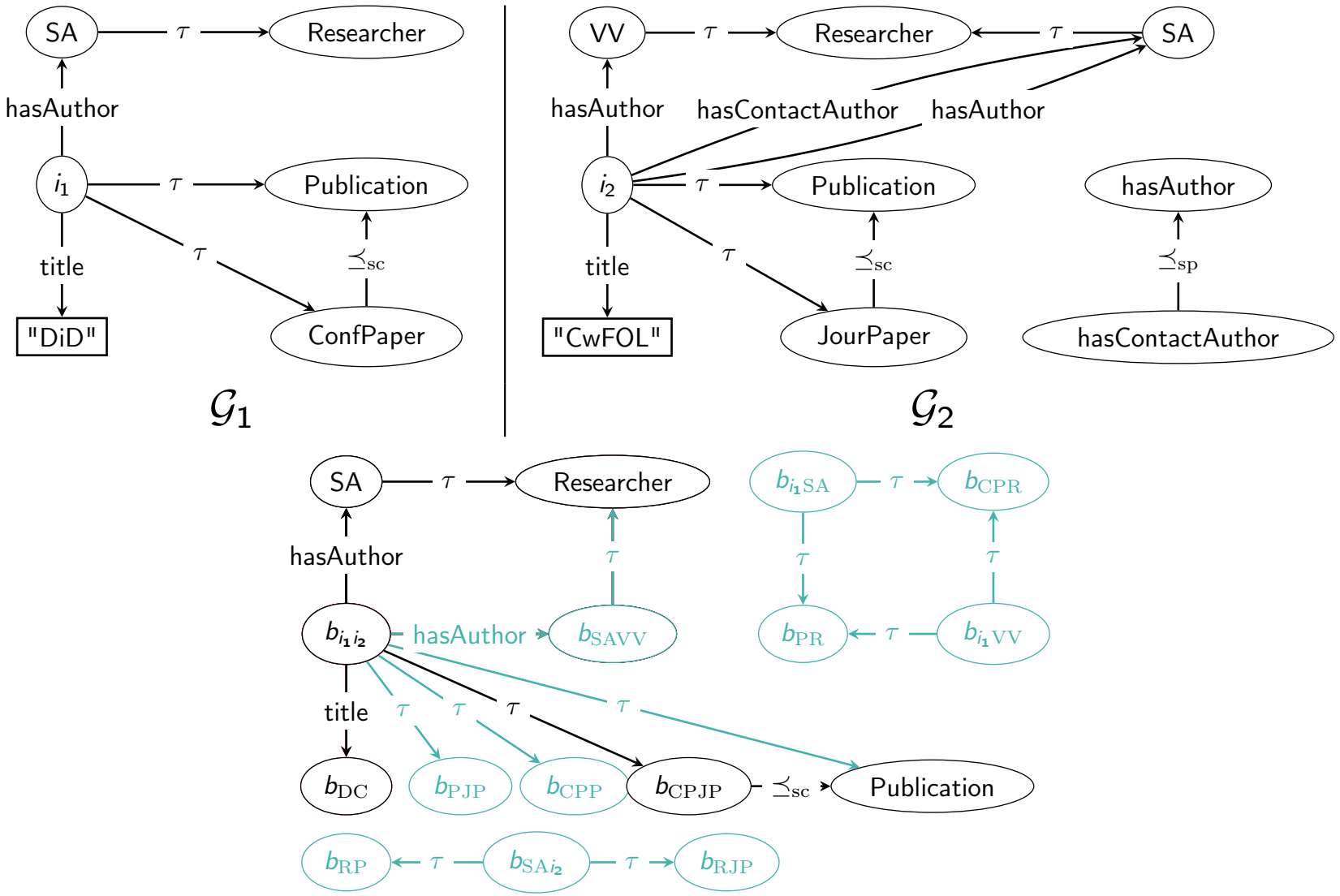
The cover graph of RDF graphs



The cover graph of RDF graphs



The cover graph of RDF graphs



Cover graph vs 1gg

Theorem ($\mathcal{R} = \emptyset$)

The *cover graph* $\mathcal{G}$ of the RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ is an 1gg of them for the empty set $\mathcal{R}$ of RDF entailment rules (i.e., $\mathcal{R} = \emptyset$).

Cover graph vs lgg

Theorem ($\mathcal{R} = \emptyset$)

The *cover graph* $\mathcal{G}$ of the RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ is an lgg of them for the empty set $\mathcal{R}$ of RDF entailment rules (i.e., $\mathcal{R} = \emptyset$).

Proposition ($\mathcal{R} = \emptyset$)

The cover graph of two RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ can be computed in $O(|\mathcal{G}_1| \times |\mathcal{G}_2|)$; its size is bounded by $|\mathcal{G}_1| \times |\mathcal{G}_2|$.

Cover graph vs lgg

Theorem ($\mathcal{R} = \emptyset$)

The *cover graph* $\mathcal{G}$ of the RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ is an lgg of them for the empty set $\mathcal{R}$ of RDF entailment rules (i.e., $\mathcal{R} = \emptyset$).

Proposition ($\mathcal{R} = \emptyset$)

The cover graph of two RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ can be computed in $O(|\mathcal{G}_1| \times |\mathcal{G}_2|)$; its size is bounded by $|\mathcal{G}_1| \times |\mathcal{G}_2|$.

Theorem ($\mathcal{R} \neq \emptyset$)

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two RDF graphs, and $\mathcal{R}$ a set of RDF entailment rules. The *cover graph* $\mathcal{G}$ of $\mathcal{G}_1^\infty$ and $\mathcal{G}_2^\infty$ is an lgg of $\mathcal{G}_1$ and $\mathcal{G}_2$.

Cover graph vs lgg

Theorem ($\mathcal{R} = \emptyset$)

The *cover graph* $\mathcal{G}$ of the RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ is an lgg of them for the empty set $\mathcal{R}$ of RDF entailment rules (i.e., $\mathcal{R} = \emptyset$).

Proposition ($\mathcal{R} = \emptyset$)

The cover graph of two RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ can be computed in $O(|\mathcal{G}_1| \times |\mathcal{G}_2|)$; its size is bounded by $|\mathcal{G}_1| \times |\mathcal{G}_2|$.

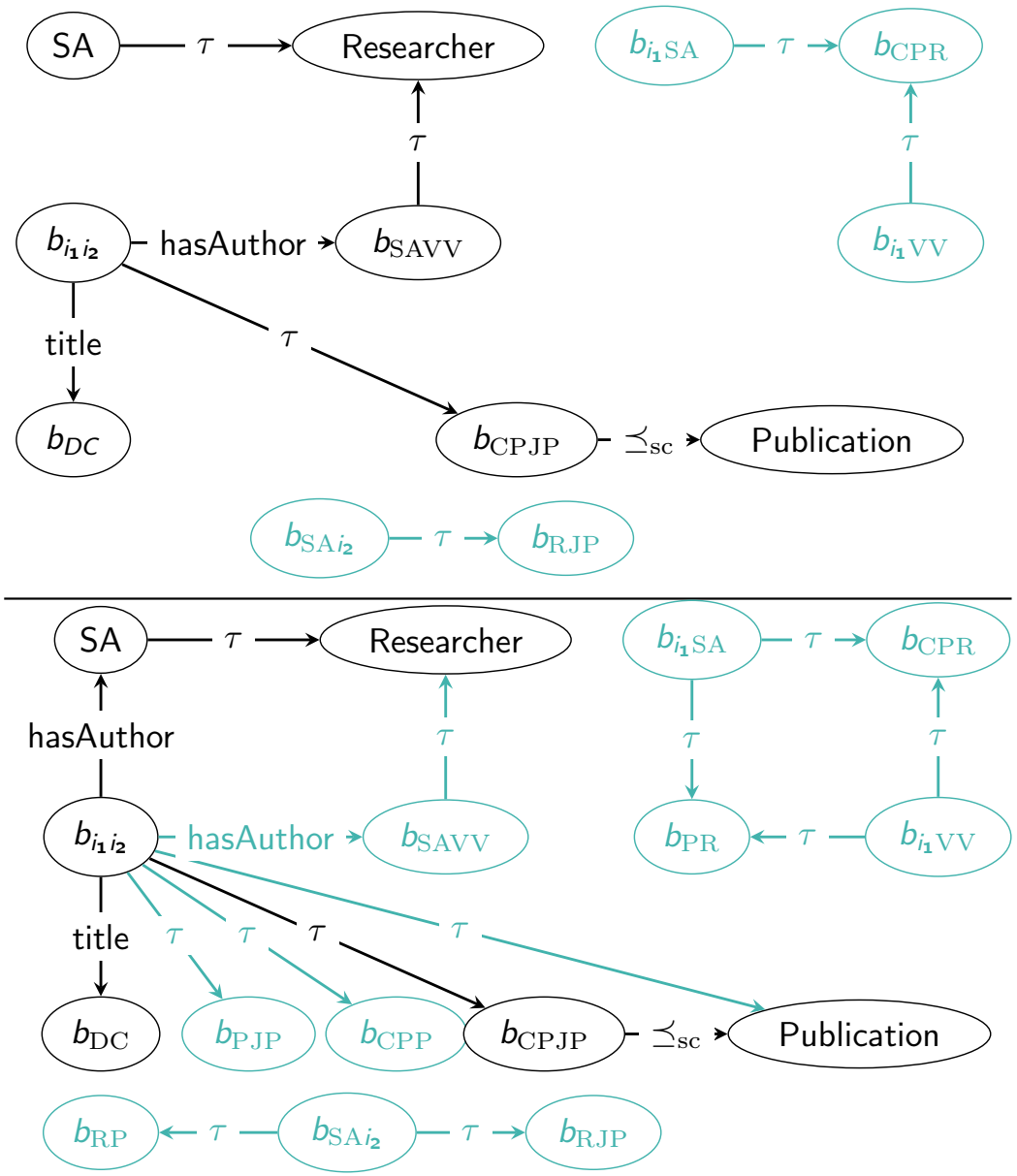
Theorem ($\mathcal{R} \neq \emptyset$)

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two RDF graphs, and $\mathcal{R}$ a set of RDF entailment rules. The *cover graph* $\mathcal{G}$ of $\mathcal{G}_1^\infty$ and $\mathcal{G}_2^\infty$ is an lgg of $\mathcal{G}_1$ and $\mathcal{G}_2$.

Corollary ($\mathcal{R} \neq \emptyset$)

An lgg of two RDF graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ can be computed in $O(|\mathcal{G}_1^\infty| \times |\mathcal{G}_2^\infty|)$ and its size is bounded by $|\mathcal{G}_1^\infty| \times |\mathcal{G}_2^\infty|$.

Cover graph vs lgg



Outline

Related work

Structural based approach

- Description Logics $\mathcal{EL}$
  - F. Baader and al. :*Computing least common subsumers in description logics with existential restrictions*.In IJCAI, 1999.
  - B. ZarrieB and al. :*Most specific generalizations w.r.t. general EL-TBoxes*.In IJCAI, 2013.
- RDF
  - SPARQL : tree queries
    - J. Lehmann and L. Buhmann. *Autosparql : Let users query your knowledge base*. In ESWC, 2011.
  - Rooted graphs, ignore RDF entailment :
    - S. Colucci and al. :*Defining and computing least common subsumers in RDF*.J. Web Semantics, 39(0), 2016.

Independent structure approach

- Conceptual Graphs
  - M. Chein and M. Mugnier.*Graph-based Knowledge Representation - Computational Foundations of Conceptual Graphs*.Springer, 2009.

Conclusion

- Revisit the problem of computing a least general generalization in the entire setting of RDF.
- Algorithms to compute `lggs` of small-to-huge RDF graphs.
  - Memory
  - Data management system
  - MapReduce
- Perspective : Heuristics in order to compute `lgg` without redundants triples.

Thank you !

Questions ?

References I

[1] F. Baader, R. Kiisters, and R. Molitor.
Computing least common subsumers in description logics with existential restrictions.
In *IJCAI*, 1999.

[2] F. Baader, B. Sertkaya, and A.-Y. Turhan.
Computing the least common subsumer w.r.t. a background terminology.
Journal of Applied Logic, 5(3), 2007.

[3] M. Chein and M. Mugnier.
Graph-based Knowledge Representation - Computational Foundations of Conceptual Graphs.
Springer, 2009.

[4] S. Colucci, F. Donini, S. Giannini, and E. D. Sciascio.
Defining and computing least common subsumers in RDF.
J. Web Semantics, 39(0), 2016.

[5] S. Colucci, F. M. Donini, and E. D. Sciascio.
Common subsumbers in RDF.
In *AI*IA*, 2013.

[6] J. Lehmann and L. Bühmann.
Autosparql : Let users query your knowledge base.
In *ESWC*, 2011.

[7] RDF 1.1 semantics.
https://www.w3.org/TR/rdf11-mt/.

[8] B. Zarrieß and A. Turhan.
Most specific generalizations w.r.t. general EL-TBoxes.
In *IJCAI*, 2013.