Refining Transitive and Pseudo-Transitive Relations at Web Scale

Shuai Wang, Joe Raad, Peter Bloem, Frank van Harmelen

KR&R Group, Vrije Universiteit Amsterdam

LISN, University of Paris-Saclay
24th January, 2022

Content

Introduction

- `(R, rdf:type, owl:TransitiveProperty)`.
- examples of transitive relations:
  - `rdfs:subClassOf` and `rdfs:subPropertyOf`
  - `dbo:previousWork` and `dbo:subsequentWork`
  - `dbo:isPartOf` and `dc:hasPart`
  - `dbo:predecessor` and `dbo:successor`
  - `prov:wasDerivedFrom`
  - `dc:creatorOf`
  - dependency, causality, subsequent event, ownership, etc.

Introduction

- *pseudo-Transitive* relations: intended to be transitive and anti-symmetric, even though not formally asserted.
- Transitivity + cycles = confusion + errors.

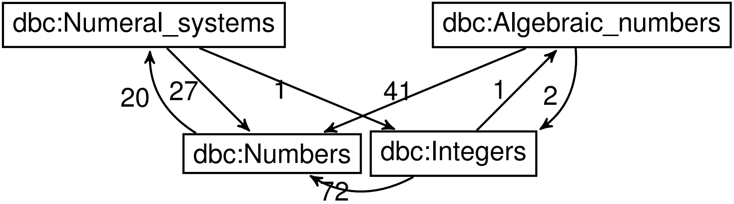


Figure 1.1: An example subgraph of `skos:broader` with weights.

Transitive Relations in the LOD Cloud

- LOD-a-lot is the integrated result of 650K datasets in the LOD Laundromat, a crawl of the LOD cloud.
- 2,486 transitive relations $\approx$ 2.7% of all triples.
- Closure under `owl:inverseOf` and `rdfs:subPropertyOf`.
- 8,804 relations in closure $\approx$ 19.5% of all triples.
- Investigate only 10 popular (pseudo-)transitive relations.
- Exclude `owl:sameAs` and `foaf:knows`.

The Problem

Issue often not a directed acyclic graph (DAG).

Task remove as few edges as possible to make it acyclic.

Complexity = Minimum Weighted Feedback Arc Set (MWFAS) problem in graph theory. It's APX-hard.

Intuition nested cycles suggest erroneous edges.

Related Work 1

- Refinement of domain-specific graphs with an intended hierarchical structure, e.g. that of `rdfs:subClassOf`.
- Often first infer a graph hierarchy and then use the pre-defined roots for one dataset [Sun et al.].
- Integrated graphs at web-scale with very limited efficiency and scalability [Wang et al.].
- Wikipedia category graph and requires external information in English [Paulheim et al.].

Related Work 2

- Depth-first traversal (DFS): removes all arcs that form a cycle while doing depth-first traversal. Complexity $\mathcal{O}(m+n)$.
- Greedy (GRD): greedy search with "sinks" and "sources". Complexity: $\mathcal{O}(m+n)$.
- KwikSort (KS): quick sort. Complexity: $\mathcal{O}(n \log n)$.
- BergerShor(BS): starts with a random permutation and compares the in-degree and out-degree of the vertices. Complexity: $\mathcal{O}(m+n)$.

The Challenge

- A very large integrated graph (web-scale)
- Cross-dataset/domain/namespace
- Multilingual
- No hierarchy - No root node
- Complex nested cycles
- No manually annotated datasets for evaluation
- No measure or statistics reported

Hypotheses

Hypothesis 1

By considering **graph structural properties**, we can remove fewer edges than general-purpose graph theoretical methods.

> graph structural properties := how edges are involved in complex nested cycles

Hypothesis 2

Using the reliability of triples (weights), we can improve the accuracy of identifying erroneous edges.

Why Measures?

- $|E|$ and $|V|$ tells us how big, but not how complex.
- Existing measures: Average Clustering Index, Global Reaching Centrality, etc.

Why Measures?

- Q1: Where are the cycles?
- Q2: How complex are they?

Answer for Q1: Strongly Connected Components

- A Strongly Connected Component (SCC) is a subgraph where any two of its vertices can be reached by a path and is maximal for this property.
- an SCC = the maximal subgraph that is strongly connected.

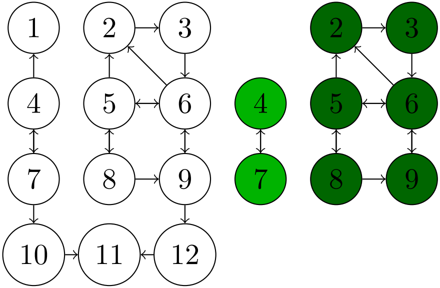


Figure 3.1: An example graph (G) and its SCCs.

SCCs for Measures

Table 1: SCCs for measures

Graph	#Edges	#Vertices	#Edges of SCCs	#Vertices of SCCs
skos:broader	11.8m	5.7m	356.9k	82.0k
skos:narrower	817.1k	737.3k	48	24
rdfs:subClassOf	4.4m	3.6m	1.4k	837

OMG! We can neglect a lot of edges!

Answer for Q2: New measures

	Easy Cases	Harder Cases
Graph Property	size-two cycles	longer chains or nested
Reason of Cycle	direction of relation	other reasons

Intuition: proportion of the easy ones and the hard ones, respectively.

Answer for Q2: New measures

Alpha measure

- numerator: #edges in cycles of size two.
- denominator: #edges in its SCCs.

Beta measure

First, remove cycles of size two from G to get G'.

- numerator: #edges remain in the SCCs of G'.
- denominator: #edges in its SCCs.

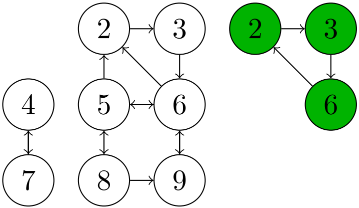
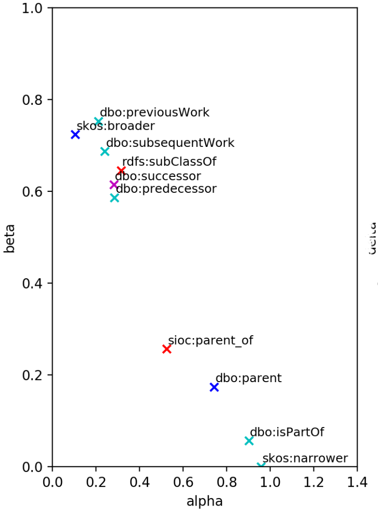


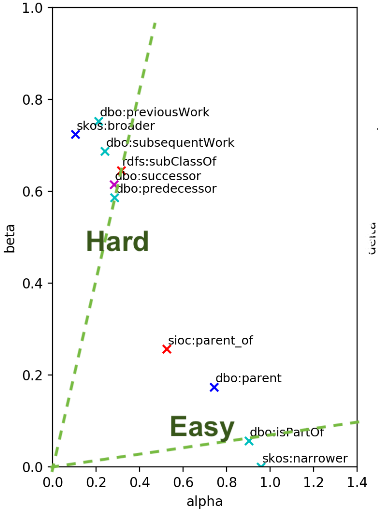
Figure 3.2: The SCCs of G and the SCCs of G'.

Answer for Q2: New measures



Cycles of `skos:narrower` can be resolved by simply making decisions on pairs of entities.
The problem of it's inverse, `skos:broader`, is much harder!

Gamma-Delta measure



Gamma and delta estimate the effort required to make a graph cycle-free.

Algorithm

We propose a divide-and-conquer algorithm:

- Divide the SCCs into partitions (if too big).
- Identify edges to remove.
- Remove identified edges and compute new SCCs.
- Repeat until all the cycles are resolved.

Algorithm: Graph partitioning

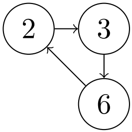
- SMT solvers can not handle some big SCCs.
- In graph theory: k-cut problem.
- There is an efficient algorithm.
- Designed two strategies for partitioning.

Algorithm: Sampling Cycles

- Face a **combinatorial explosion** when listing all cycles from a graph.
- Designed two strategies for the sampling of cycles.
- A bounded number of cycles for each round.

Algorithm: Resolving Cycles

- Encode constraints to an SMT solver.
- Introduce a propositional variable for each edge $p_{(v_i,v_j)}$.
  1. A (hard) clause for each cycle $v_1, \ldots .v_k$: $[\neg p_{(v_1,v_2)} \vee \ldots \vee \neg p_{(v_{k-1},v_k)} \vee \neg p_{(v_k,v_1)}]$.
  2. A (soft) clause $[p_{(v_i,v_j)}]$ for each edge e with its weight.
- Goal: maximum weight in bounded time
- We remove all the edges whose propositional variables are False.



Weights

- Counted weights: count the number of relevant datasets for an edge in the LOD Laundromat.
- Inferred weights: use logical redundancy.

Weights

Inferred weight for `(A, rdfs:subClassOf, B)`:

- weight 2 if also `(A, owl:equivalentClass, B)` or its reverse.
- weight 1 otherwise.

Inferred weight for `(S skos:broader T)`:

- weight 2 if also `(T, skos:narrower, S)`.
- weight 1 otherwise.

Gold Standard

- Manually annotated triples in the subgraphs regarding `rdfs:subClassOf` and `skos:broader`.
- G1: randomly pick 500 edges.
- G2: a variant way that splits that of size two with the rest (200+500 edges).
- Only edges in SCCs and neglected the rest.
- We developed a tool for annotation, namely ANNit.

Implementation and Experimental Setting

- Introduction
  - The Problem
- Related Work
  - Hypotheses
- Measures
- Algorithm
- Gold Standard
- **Implementation**
- Evaluation
- Contributions

- Two partitions at each step, i.e. k = 2.
- 3,000 cycles obtained at each step.
- Tarjan, Pymetis, Z3.
- SMT's time bound = 10 seconds.

Evaluation: Hypothesis 1

Relations	Scope	BS	GRD	KS	DFS	Approach
skos:broader	Overall	1m	493k	5m	125k	**114k**
	In SCCs	327k	356k	177k		
rdfs:subclassOf	Overall	4m	25k	219.2	529	**330**
	in SCCs	1k	430	716		
dbo:isPartOf	Overall	18k	2,175	459k	2,286	**2,143**
	In SCCs	3k	2,153	2,331		
dbo:successor	Overall	85k	24k	218k	17k	**13k**
	In SCCs	43k	17k	29k		

Supports Hypothesis 1: taking the graph structure into account, we removed the least amount of edges.

Evaluation: Hypothesis 2

Relations	BS	GRD	KS	DFS	Approach with counted weights
skos:broader	0.32	0.42	0.33	0.35	**0.44**
rdfs:subclassOf	0.40	0.42	0.38	0.43	**0.53**

Supports Hypothesis 2: using weights can improve the precision for `skos:broader` and reduce the removed edges.

List of Contributions

1. **Measures**: The Alpha-Beta and Gamma-Delta measures.
2. **Algorithm**: a generic scalable approach for the refinement of (pseudo-)transitive relations using an SMT solver by exploiting Strongly Connected Components.
3. **Results**: our results support our hypotheses.
4. **Datasets**: a dataset of ten (pseudo-)transitive relations with weights.
5. **Gold standard**: thousands of manually annotated triples.

Thank You for your attention!

Contact: shuai.wang@vu.nl

Discussion

- Edges removed during graph partitioning.
- Unstable results for `rdfs:subClassOf`.
- Counted weights are better than inferred weights.
- P2S1 is the suggested parameter setting when weights are present.
- good precision, bad recall.

Discussion

- The weights of `skos:broader` follow a power-law distribution.