Querying (Knowledge) Graphs (vs. Graph Learning and LLMs?)

Recap of some past work and how it could connect to the present

Axel Polleres

Vienna University of Economics and Business (WU)
Institute for Data, Process and Knowledge Management

The presented research was funded in whole or in part by the Austrian Science Fund (FWF) [10.55776/COE12]

Outline

- Background: Collaborative KGs
- **Efficient KG Archiving and Querying** with HDT [1]
- Applications:
  - Making SPARQL endpoints more efficient [2,3]
  - Making Message-Passing-based Graph processing more efficient [4]
- Open Questions … many ☺

Motivation

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

In our group we have done a lot of work on

- *querying*,
- *analysing*, and
- *improving* the quality

of **Collaborative KGs at scale**

Collaborative, Open Knowledge Graphs: DBpedia

WU WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

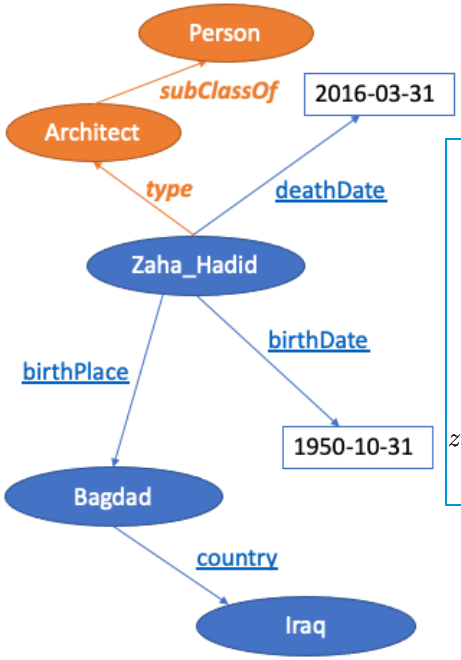
DBpedia

DBpedia generates a graph from links and facts in Wikipedia's Infoboxes:

http://wikipedia.org/wiki/Zaha_Hadid

Born	Zaha Mohammad Hadid 31 October 1950 Baghdad, Kingdom of Iraq
Died	31 March 2016 (aged 65) Miami, Florida, U.S.
Nationality	Iraq, United Kingdom
Alma mater	American University of Beirut Architectural Association School of Architecture
Occupation	Architect
Parent(s)	Mohammed Hadid Wajeeha Sabonji
Practice	Zaha Hadid Architects
Buildings	Vitra Fire Station, MAXXI, Bridge Pavilion, Contemporary Arts Center, Heydar Aliyev Center, Riverside Museum
Website	www.zaha-hadid.com

http://dbpedia.org/resource/Zaha_Hadid



RDF+OWL:

Data + "Ontology" *(i.e. class and property hierarchy)* both encoded in a directed labeled graph:

$$zaha\_hadid \in Architect$$

$$Architect \sqsubseteq Person$$

EFMD EQUIS ACCREDITED AACSB ACCREDITED ASSOCIATION AMBA ACCREDITED

Collaborative, Open Knowledge Graphs: Wikidata

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

WIKIDATA

Lionel Messi (Q615)

Argentine association football player

image	
occupation	association football player 2 references
FIFA player ID (archived)	229397 1 reference
country of citizenship	Argentina start time 1987 0 references
	Spain start time 2005 1 reference

Revision history of "Lionel Messi" (Q615)

View logs for this item (view abuse log)

Filter revisions

Diff selection: Mark the radio buttons of the revisions to compare and hit enter or the button at the botto

Legend: **(cur)** = difference with latest revision, **(prev)** = difference with preceding revision, **m** = minor ec

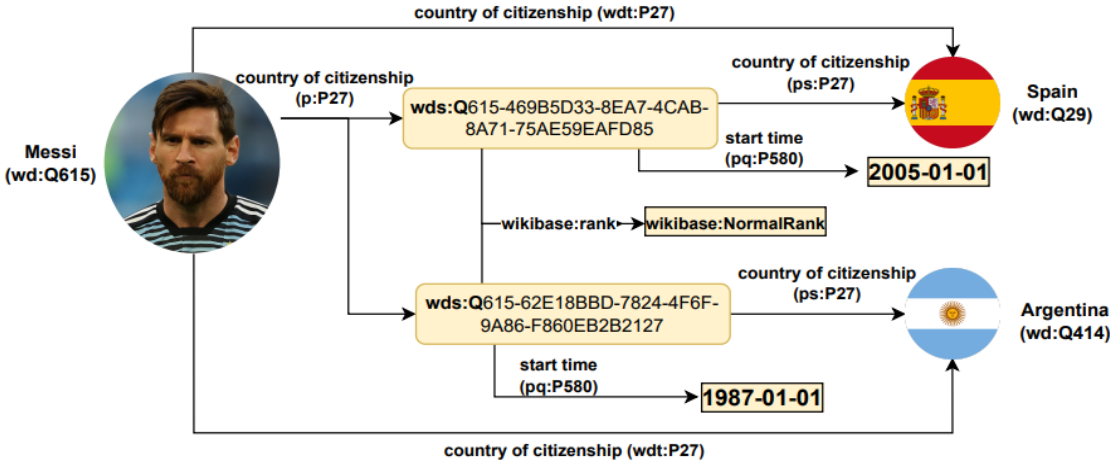
(latest | earliest) View (newer 50 | older 50) (20 | 50 | 100 | 250 | 500)

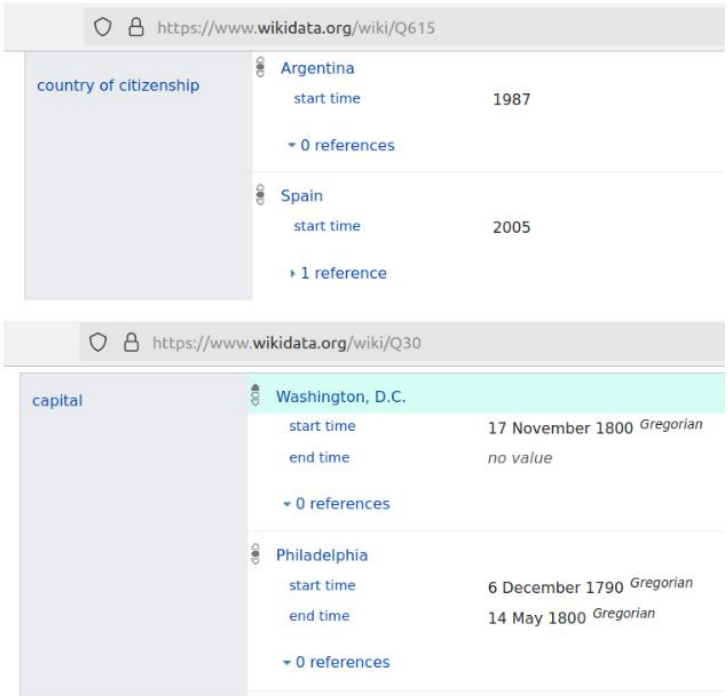
Compare selected revisions

- (cur | prev) 08:56, 8 December 2024 দ্রাদ্রুল নাজা (talk | contribs) . . (537,664 bytes) (+92) . .
- (cur | prev) 20:39, 7 December 2024 Sanremofilo (talk | contribs) . . (537,572 bytes) (+363) *messi/5663)* (Tag: Wikidata user interface)
- (cur | prev) 15:21, 2 December 2024 Ytterbyz (talk | contribs) . . (537,209 bytes) (+349) . . (Tag: Wikidata user interface)
- (cur | prev) 23:45, 29 November 2024 Mickey Đại Phát (talk | contribs) . . (536,860 bytes) (- Wikidata user interface, Mobile termbox)
- (cur | prev) 19:47, 27 November 2024 KrBot (talk | contribs) . . (536,877 bytes) (−14) . . *(Se see autofix на / on Property talk:P12924)*

Expressing everthing as a labelled graph Wikidata's proprietary reification model

- Wikidata's internal Data Model, which consits of claims with additional context indormation is fitted into a "flat" RDF (directed labelled graph/triples) model:

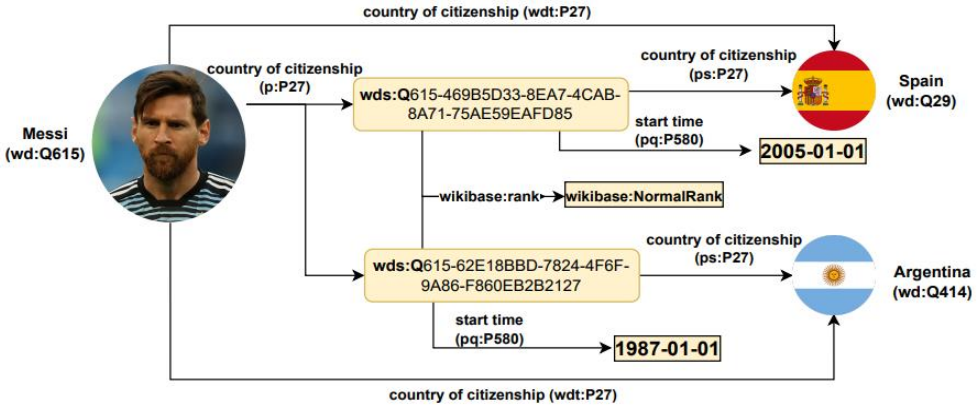




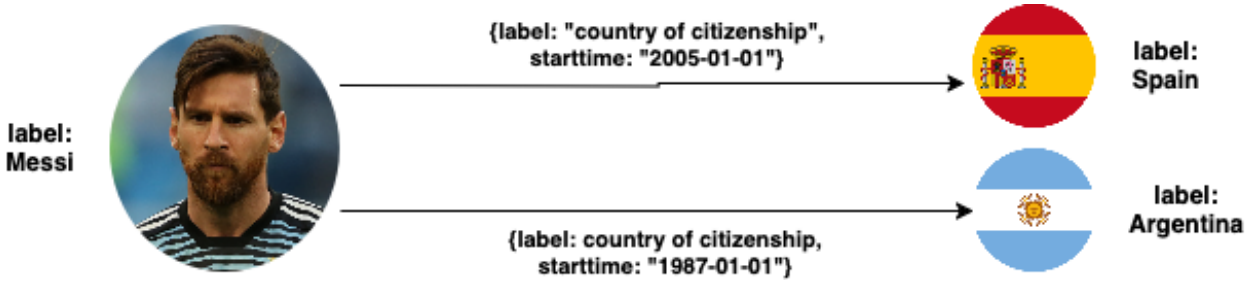
See our recent ISWC2024 tutorial:
https://ww101.ai.wu.ac.at/

Not all Graphs are equal: Reification

- i.e. this is just an ugly "reified" form in RDF…



  - … of what people often call a Labelled Property Graph (LPG)*:



** Side note: For more details on different (Knowledge) Graph data models and constraint languages see [8]*

Scale: Let's have a look at practical examples of such collaboratively curated Knowledge Graphs:

- DBPedia (since 2007) vs. Wikidata (since 2012)

DBpedia

DBpedia

Developer(s)	Leipzig University University of Mannheim
Initial release	10 January 2007 (17 years ago)
Stable release	DBpedia 2016-10 / 4 July 2017
Repository	github.com/dbpedia/
Written in	Scala · Java
Type	Semantic Web · Linked Data
License	GNU General Public License
Website	dbpedia.org

✓	RDF	✓
✓	SPARQL (standard QL for RDF) endpoint	✓
✓	Standard Ontology Language (OWL)	✗
✗	Consistent	✗
✗	Context	✓

Wikidata

WIKIDATA

Screenshot	[show]
Type of site	Knowledge base · Wiki
Available in	Multiple languages
Owner	Wikimedia Foundation
Editor	Wikimedia community
URL	www.wikidata.org/wiki/Wikidata:Main_Page
Commercial	No
Registration	Optional
Launched	29 October 2012; 12 years ago[1]

58.167.851 #subjects/nodes
68.687 #properties
1.040.358.853 #triples/edges

1.790.689.565 #subject (117,467,468 #nodes)
60.099 #properties
106.962 #classes
8.348.213.968 #triples/edges

vs. Status of Graph learning?

Text.	Data	Year	Task	# Nodes	# Edges	Domain	Source & Notes
Node	ogb-arxiv	2020.5	NC	169,343	1,166,243	Academic	OGB [186]
	ogb-products	2020.5	NC	2,449,029	61,859,140	E-commerce	OGB [186]
	ogb-papers110M	2020.5	NC	111,059,956	1,615,685,872	Academic	OGB [186]
	ogb-citation2	2020.5	LP	2,927,963	30,561,187	Academic	OGB [186]
	Cora	2000	NC	2,708	5,429	Academic	[10]
	Citeseer	1998	NC	3,312	4,732	Academic	[11]
	DBLP	2023.1	NC, LP	5,259,858	36,630,661	Academic	www.aminer.org/citation
	MAG	2020	NC, LP, Rec RG	~ 10M	~ 50M	Academic	multiple domains [12] [13]
	Goodreads-books	2018	NC, LP	~ 2M	~ 20M	Books	multiple domains [14]
	Amazon-items	2018	NC, LP, Rec	~ 15.5M	~ 100M	E-commerce	multiple domains [15]
	SciDocs	2020	NC, UAP, LP, Rec	-	-	Academic	[50]
	PubMed	2020	NC	19,717	44,338	Academic	[16]
	Wikidata5M	2021	LP	~ 4M	~ 20M	Wikipedia	[17]
	Twitter	2023	NC, LP	176,279	2,373,956	Social	[52]
Edge	Goodreads-reviews	2018	EC, LP	~ 3M	~ 100M	Books	multiple domains [14]
	Amazon-reviews	2018	EC, LP	~ 15.5M	~ 200M	E-commerce	multiple domains [15]
	Stackoverflow	2023	EC, LP	129,322	281,657	Social	[73]

Large Language Models on Graphs: A Comprehensive Survey (Jin et al. 2023)
https://arxiv.org/abs/2312.02783

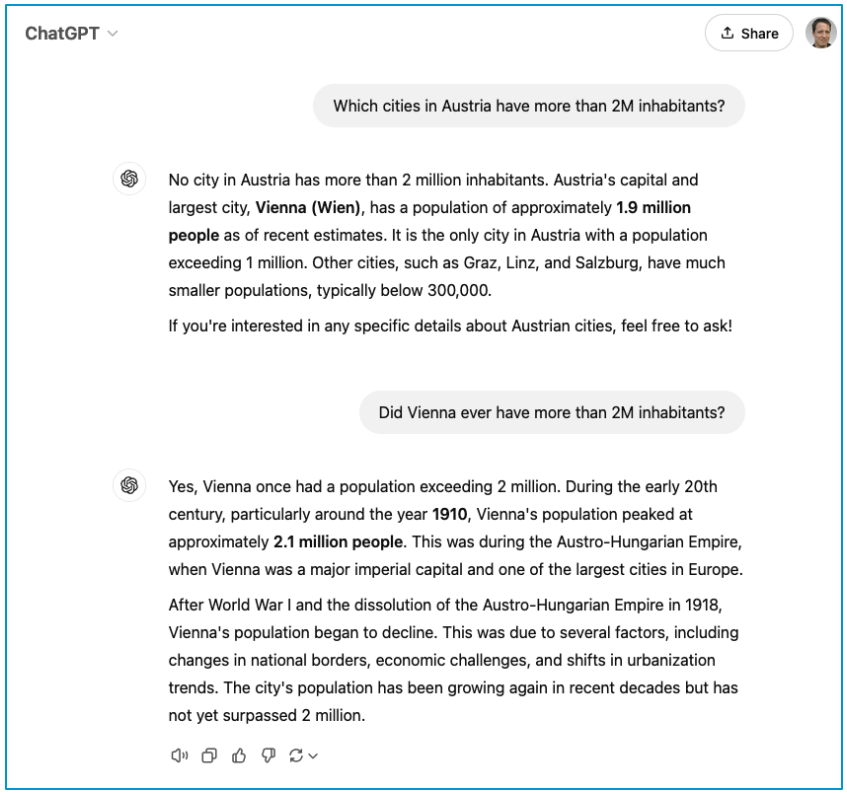
Open Graph Benchmark: Datasets for Machine Learning on Graphs (Hu et al. 2020)
https://arxiv.org/abs/2005.00687

https://ogb.stanford.edu/docs/linkprop/#ogbl-wikikg2
Wikidata extract with 539 relation types…

So, for what are these KGs actually good for in the age of LLMs and AI?

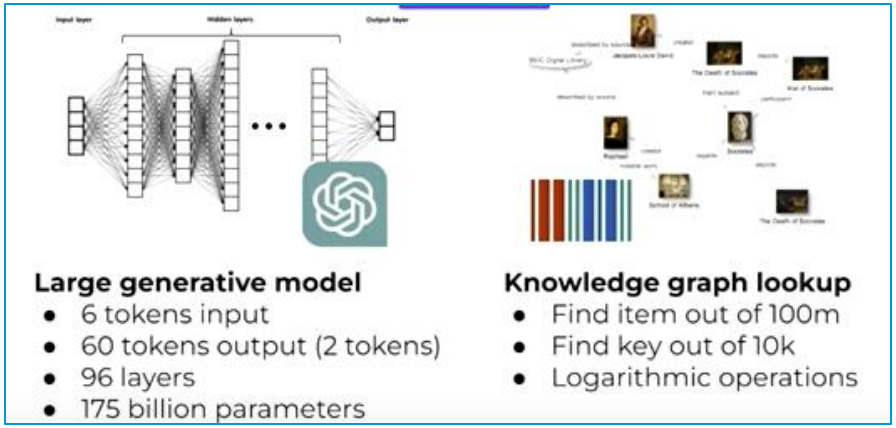
WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Main task:
Complete, factually reliable, efficient query answering



- KGC23 Keynote: "The Future of Knowledge Graphs in a World of LLMs — Denny Vrandečić, Wikimedia"

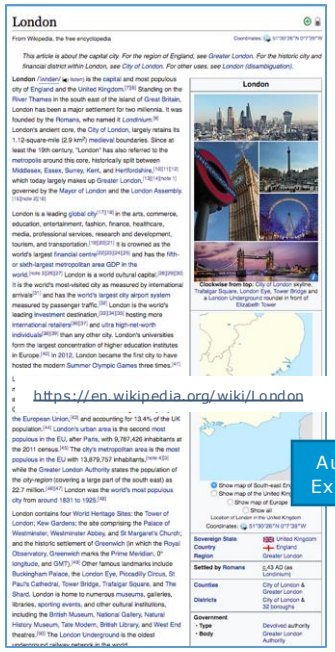
https://www.youtube.com/watch?v=ww99npDh4cg



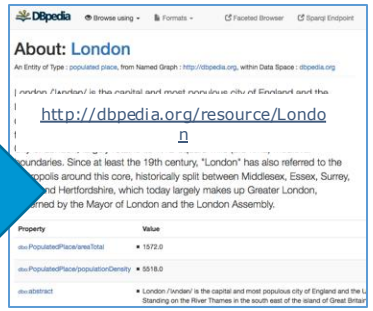
EFMD EQUIS ACCREDITED AACSB ACCREDITED ASSOCIATION AMBA ACCREDITED

SPARQL: Using KGs to answer questions:

- E.g. from DBpedia



Automatic Exctractors



- You can use a language called SPARQL endpoint (roughly: SQL for RDF) to do ***structured queries*** over RDF:
  - „Cities in the UK with more than 1M population":

Structured queries (SPARQL):

https://api.triplydb.com/s/qZZskqRpQ

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX yago: <http://dbpedia.org/class/yago/>

SELECT DISTINCT ?city ?pop WHERE {
   ?city a schema:City .
   ?city dbo:country :United_Kingdom.
   ?city dbo:populationTotal ?pop

   FILTER ( ?pop > 1000000 )
}
```

The same question as before in Wikidata:

Note: Wikidata does not use standard OWL

Note: Wikidata uses numeric IDs

- "Simple" surface query:

  Which cities in the UK have more than 1M people?

```
SELECT DISTINCT ?city WHERE {
                ?city wdt:P31/wdt:P279* wd:Q515.
                ?city wdt:P1082 ?population .
                ?city wdt:P17 wd:Q38 .
                FILTER (?population > 1000000) }
```

city (Q515)
large and permanent human settlement

population (P1082)
number of people inhabiting the place; number of people of subject

country (P17)
sovereign state of this item

United Kingdom (Q145)
country in Europe

- What's this?

instance of (P31)
that class of which this subject is a particular example and member. (Subject typically an individual member with Proper Name label.) Different from P279 (subclass of).

subclass of (P279)
all instances of these items are instances of those items; this item is a class (subset) of that item. Not to be confused with Property:P31 (instance of).

The same question as before in Wikidata:

https://w.wiki/BqRX

Which cities in the Austria have more than 1M/2M people?

Wikidata also has such contextual information!

```
SELECT DISTINCT ?City ?Pop
{
 ?City wdt:P17 wd:Q40;
       wdt:P31/wdt:P279* wd:Q515;
       wdt:P1082 ?Pop.
 FILTER (?Pop > 1000000)
 # note: Vienna historically had more than 2M inhabitants!
 # FILTER (?Pop > 2000000)
}
```

So, WHEN did Vienna have 2M inhabitants?

The same question as before in Wikidata:

https://w.wiki/BqRj

Which cities in the Austria have more than 1M/2M people?

```
SELECT DISTINCT ?City ?Pop ?Timepoint
{
 ?City wdt:P17 wd:Q40;
       wdt:P31/wdt:P279* wd:Q515;
       p:P1082 ?Stmnt.
 ?Stmnt ps:P1082 ?Pop;
        pq:P585 ?Timepoint.
 # FILTER (?Pop > 1000000)
 # note: Vienna historically had more than 2M inhabitants!
 FILTER (?Pop > 2000000)
}
```

So, WHEN did Vienna have 2M inhabitants? Works!!!!

But needs an understanding of **Wikidata's proprietary RDF reification model** to model context!

See our recent ISWC2024 tutorial:
https://ww101.ai.wu.ac.at/

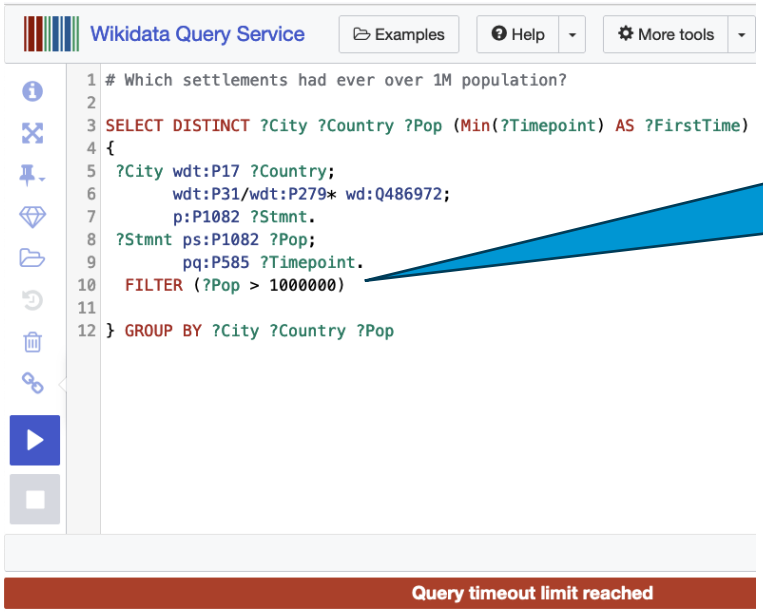
Admittedly, Denny didn't talk about this…

… at least Wikidata also struggles on some questions: https://w.wiki/CLw9

- Public query endpoints (SPARQL) hard to host

  *(**Note**: a bit like hosting/serving large LLMs to many users?)*

- Complex queries time out



Challenge: scaling queries to large-scale, schemaless KGs (complex joint, aggregations, … for many users)

For the records: comparison with GPT ;-)

https://chatgpt.com/share/675585c7-04cc-8006-a20e-c70d75619e13

Challenges:

- Many queries on DBPEdia's and Wikidata's SPARQL endpoint **time out**
- **What can we do about it?**
  - *Bespoke (Compressed) Indexing **(HDT)***
  - *Partitioning **(smart-KG)***

HDT - a “Knowledge Graph” hacker toolkit

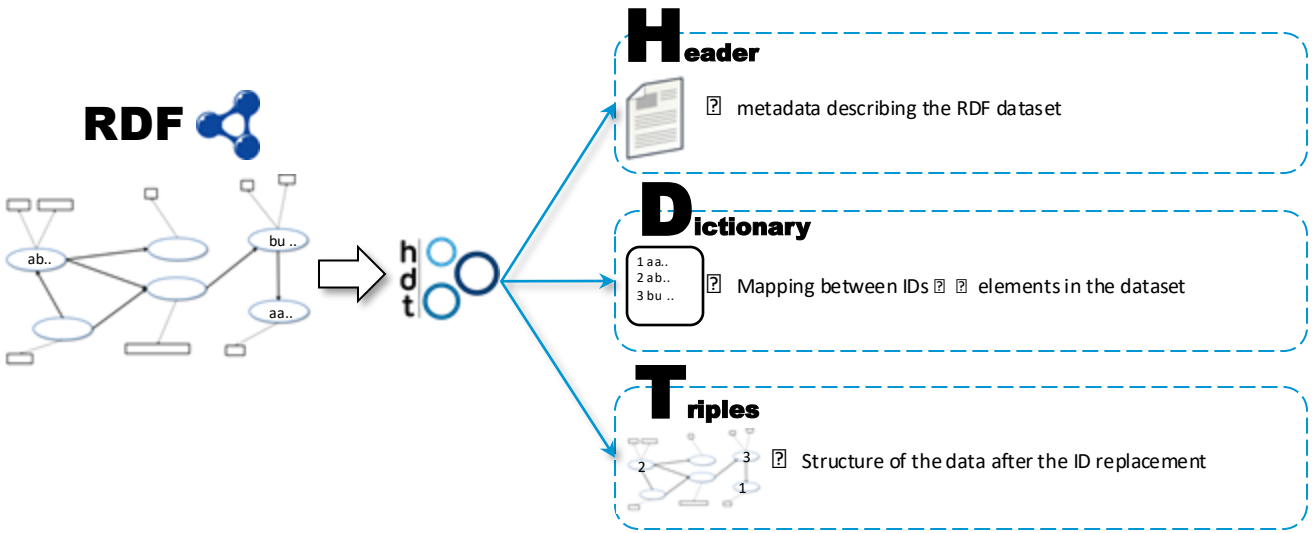
- **Compressed, queryable** serialization of RDF
- **Standardized?** W3C member submission 2011: https://www.w3.org/Submission/HDT/

- Allows fast RDF retrieval in compressed space (without prior decompression)
  - Includes internal indexes to solve basic queries with small memory footprint.
    - Very fast on basic queries (triple patterns), x 1.5 faster than Virtuoso, Jena, RDF3X.
    - Supports FULL SPARQL as the compressed backend store of **Jena**, with an efficiency on the same scale as current more optimized solutions

▷Slightly more but you can query!

- Challenges:
  - Publisher has to pay a bit of overhead to convert the RDF dataset to HDT (but then it is ready to consume efficiently!)
  - Inefficient for (live) updates … *(**Note**: another parallel to LLMs?)*

HDT (Header-Dictionary-Triples) Overview

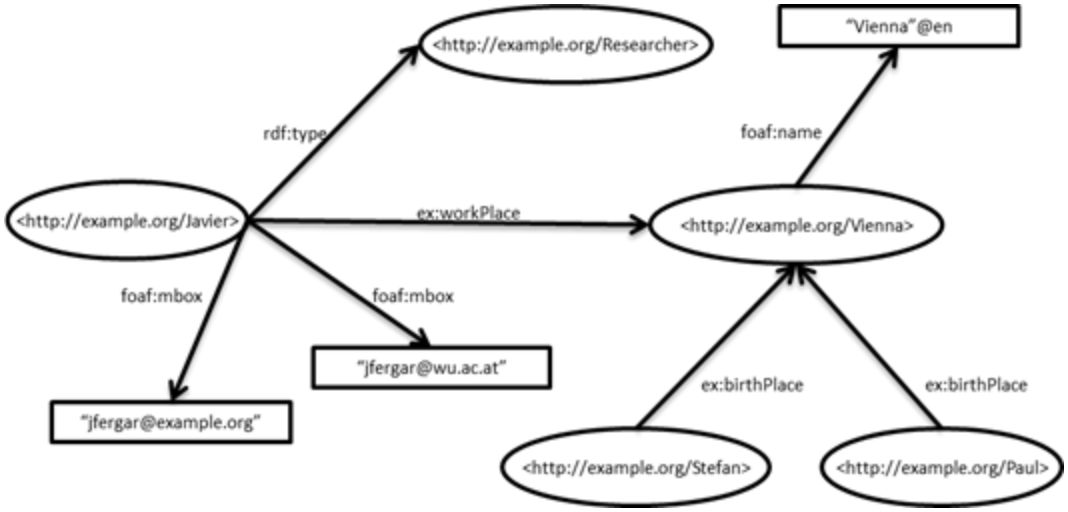


HDT – Header information:

```
$ hdtInfo wikidata20210305.hdt


<file://[latest-all.ttl.gz]> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://purl.org/HDT/hdt#Dataset> .
<file://[latest-all.ttl.gz]> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://rdfs.org/ns/void#Dataset> .
<file://[latest-all.ttl.gz]> <http://rdfs.org/ns/void#triples> "14578569927" .
<file://[latest-all.ttl.gz]> <http://rdfs.org/ns/void#properties> "38867" .
<file://[latest-all.ttl.gz]> <http://rdfs.org/ns/void#distinctSubjects> "1625057179" .
<file://[latest-all.ttl.gz]> <http://rdfs.org/ns/void#distinctObjects> "2538585808" .
<file://[latest-all.ttl.gz]> <http://purl.org/HDT/hdt#formatInformation> "_:format" .
_:format <http://purl.org/HDT/hdt#dictionary> "_:dictionary" .
_:format <http://purl.org/HDT/hdt#triples> "_:triples" .
<file://[latest-all.ttl.gz]> <http://purl.org/HDT/hdt#statisticalInformation> "_:statistics" .
<file://[latest-all.ttl.gz]> <http://purl.org/HDT/hdt#publicationInformation> "_:publicationInformation" .
_:publicationInformation <http://purl.org/dc/terms/issued> "2021-04-24T12:42Z" .
_:dictionary <http://purl.org/dc/terms/format> <http://purl.org/HDT/hdt#dictionaryFour> .
_:dictionary <http://purl.org/HDT/hdt#dictionarynumSharedSubjectObject> "1451915667" .
_:triples <http://purl.org/dc/terms/format> <http://purl.org/HDT/hdt#triplesBitmap> .
_:triples <http://purl.org/HDT/hdt#triplesnumTriples> "14578569927" .
_:triples <http://purl.org/HDT/hdt#triplesOrder> "SPO" .
_:statistics <http://purl.org/HDT/hdt#hdtSize> "159085366343" .
```

Dictionary+Triples partition

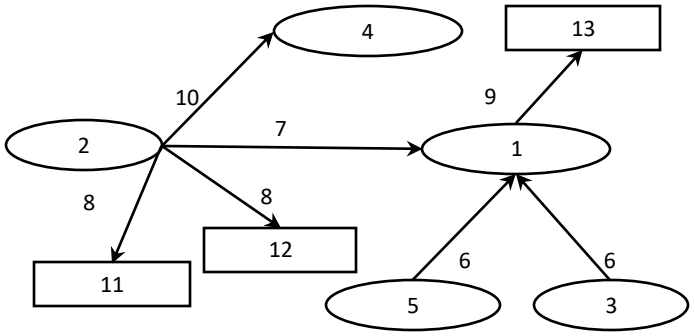


Hint: imagine HDT for now as "SPO-sorted triples"

```
ex:Vienna   foaf:name       "Vienna"@en.
ex:Javier   ex:workPlace    ex:Vienna;
            foaf:mbox       "jfergar@example.org",
                            "jfergar@wu.ac.at";
            rdf:type        ex:Researcher .
ex:Paul     ex:birthplace   ex:Vienna.
ex:Stefan   ex:birthplace   ex:Vienna.
```

Dictionary+Triples partition

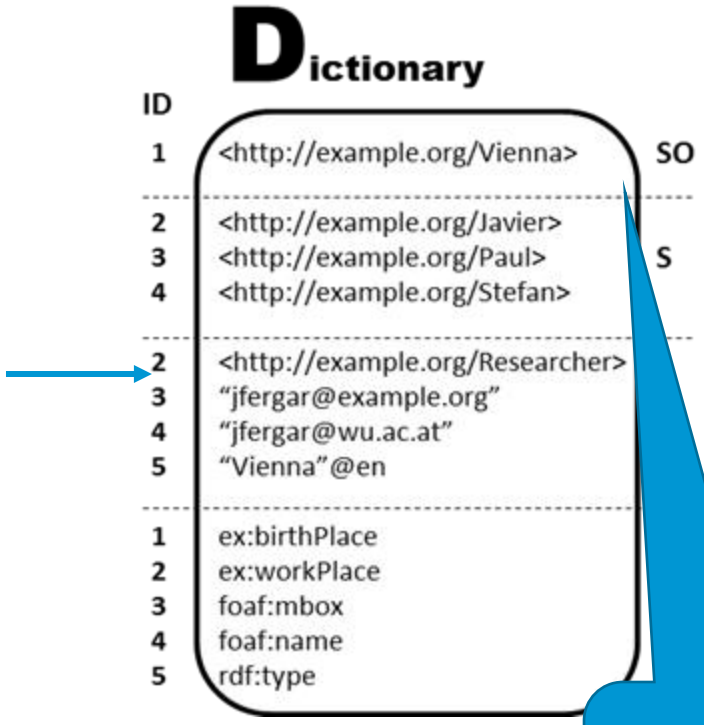
1	ex:Vienna
2	ex:Javier
3	ex:Paul
4	ex:Researcher
5	ex:Stefan
6	ex:birthPlace
7	ex:workPlace
8	foaf:mbox
9	foaf:name
10	rdf:type
11	“jfergar@example.org”
12	“jfergar@wu.ac.at”
13	“Vienna”@en



21

Dictionary (in practice)

1	ex:Vienna
2	ex:Javier
3	ex:Paul
4	ex:Researcher
5	ex:Stefan
6	ex:birthPlace
7	ex:workPlace
8	foaf:mbox
9	foaf:name
10	rdf:type
11	“jfergar@example.org”
12	“jfergar@wu.ac.at”
13	“Vienna”@en



i.e.,dictionary is not exactly "SPO-sorted" but "SO-S-O-P"-sorted

- ❿ Split by role
- ❿ Prefix-Based compression for each role
  - ❿ Efficient ID+String operations

Dictionary compression: Plain Front Coding (PFC)

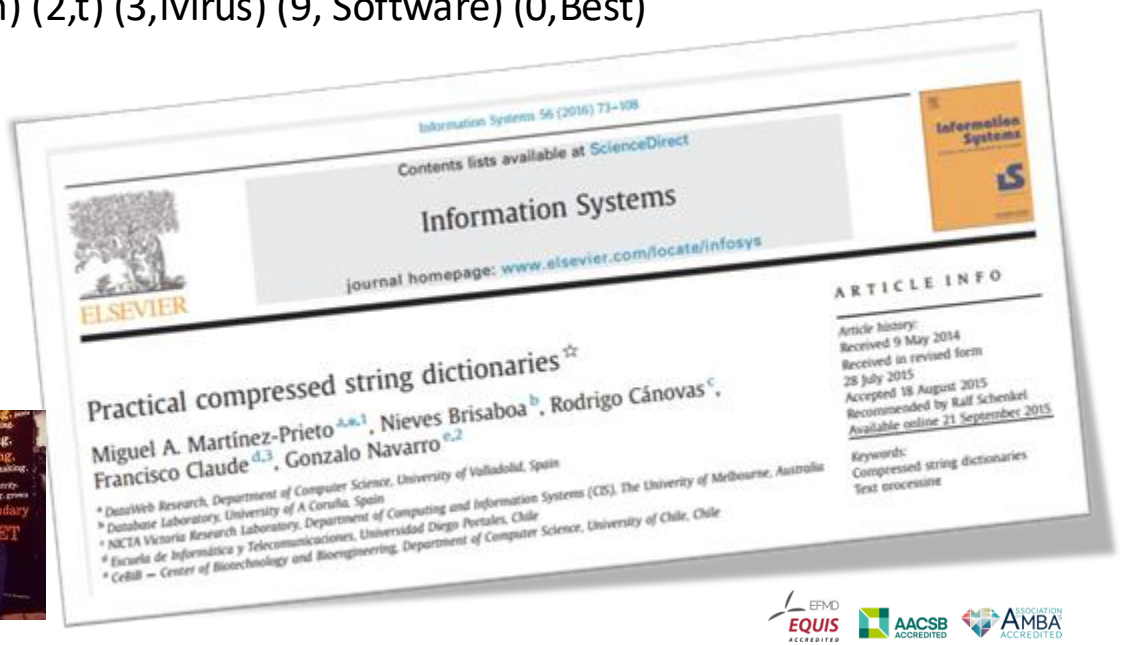
relies on prefix-based compression

Each string is encoded with two values

- An integer representing the number of characters shared with the previous string
- A sequence of characters representing the suffix that is not shared with the previous string

A
An
Ant
Antivirus
Antivirus Software
Best

➡ (0,a) (1,n) (2,t) (3,ivirus) (9, Software) (0,Best)



Information Systems 56 (2016) 73–108

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

ELSEVIER

Practical compressed string dictionaries☆

Miguel A. Martínez-Prieto[a,e,1], Nieves Brisaboa[b], Rodrigo Cánovas[c], Francisco Claude[d,3], Gonzalo Navarro[e,2]

[a] DataWeb Research, Department of Computer Science, University of Valladolid, Spain
[b] Database Laboratory, University of A Coruña, Spain
[c] NICTA Victoria Research Laboratory, Department of Computing and Information Systems (CIS), The University of Melbourne, Australia
[d] Escuela de Informática y Telecomunicaciones, Universidad Diego Portales, Chile
[e] CeBiB — Center of Biotechnology and Bioengineering, Department of Computer Science, University of Chile, Chile

ARTICLE INFO

Article history:
Received 9 May 2014
Received in revised form
28 July 2015
Accepted 18 August 2015
Recommended by Ralf Schenkel
Available online 21 September 2015

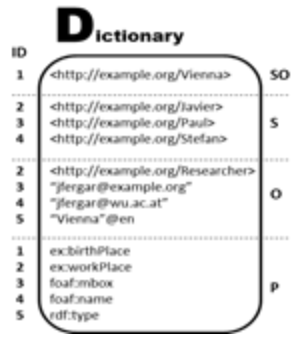
Keywords:
Compressed string dictionaries
Text processing

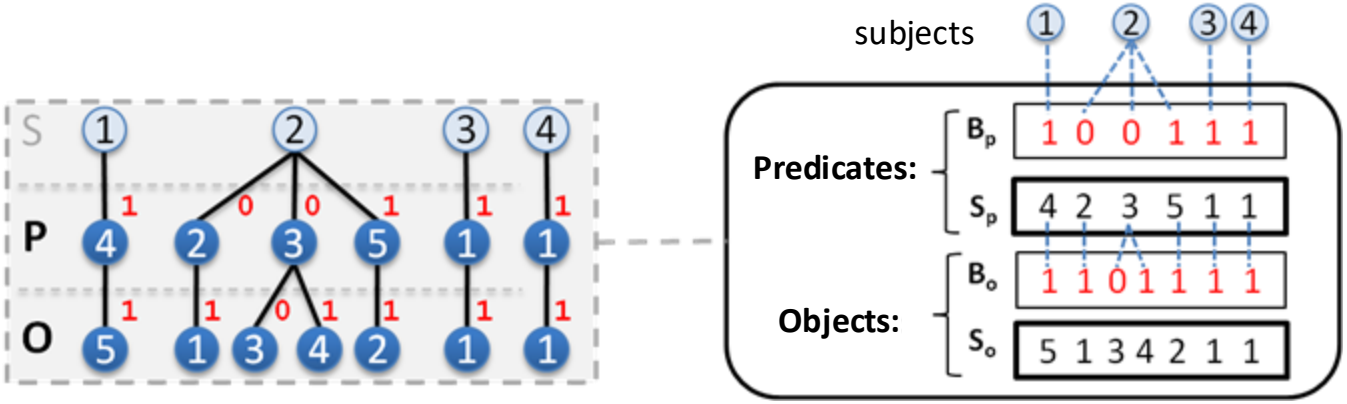
Bitmap Triples Encoding

❿ Bitmap Triples, idea a bit similar to "sorted Turtle":

```
ex:Vienna   foaf:name      "Vienna"@en.
ex:Javier   ex:workPlace   ex:Vienna;
            foaf:mbox      "jfergar@example.org",
                           "jfergar@wu.ac.at";
            rdf:type       ex:Researcher .
ex:Paul     ex:birthplace  ex:Vienna.
ex:Stefan   ex:birthplace  ex:Vienna.
```

```
1 4 5.
2 2 1;
  3 3,
    4;
  5 2.
3 1 1.
4 1 1.
```

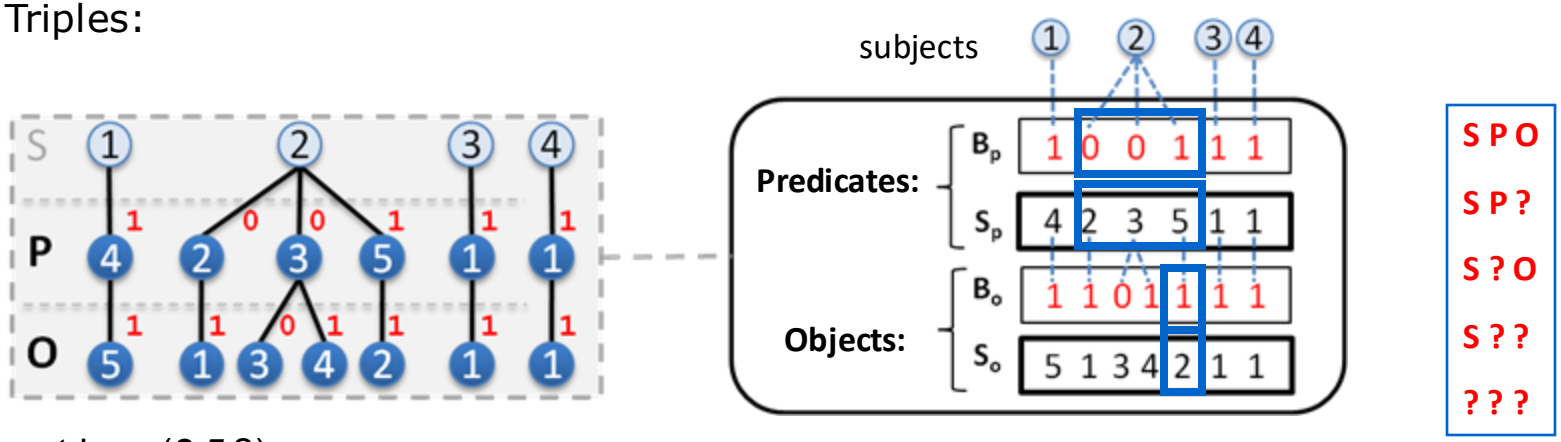




- We index the bitsequences to provide a SPO index

Bitmap Triples Encoding

❿ Bitmap Triples:



- E.g. retrieve (***2***,***5***,?)
  - Find the position of the first and **second** '1'-bits in B_p (`select`)
  - Binary search on the list of predicates S_p in this range, looking for 5
  - Note that such predicate 5 is in position 4 of S_p
  - Find the position of the **fourth** '1'-bit in B_o (`select`) -> 5th position
  - i.e. retrieve 5th value of S_o -> **2**

On-the-fly indexes: HDT-FoQ (Focus-on-Querying indexes)

- From the exchanged HDT to the functional HDT-FoQ:
  - Publish and Exchange HDT (i.e., B_p,S_p,B_o,S_o from last slide) and
  - At the consumer:

Efficient "Triple pattern" matching, i.e. "edge lookups", e.g.:

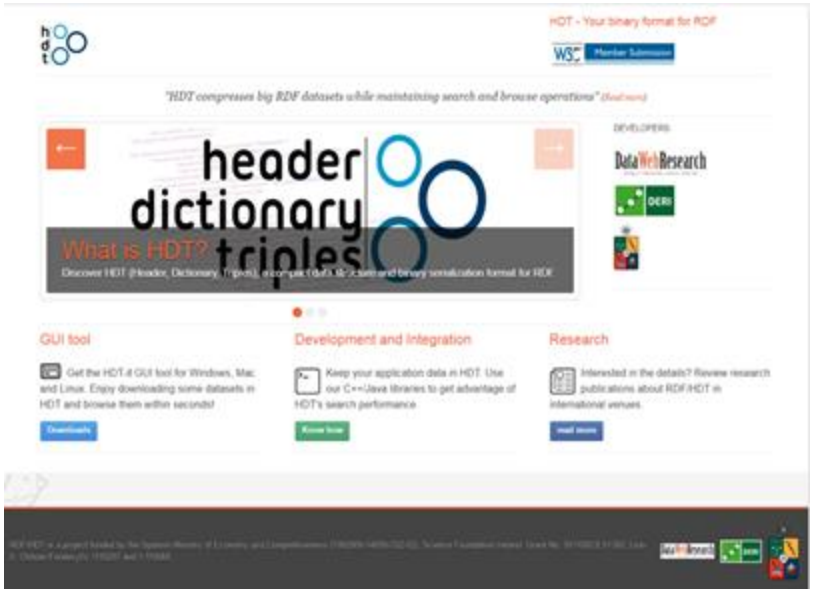
{:vienna :country ?.}

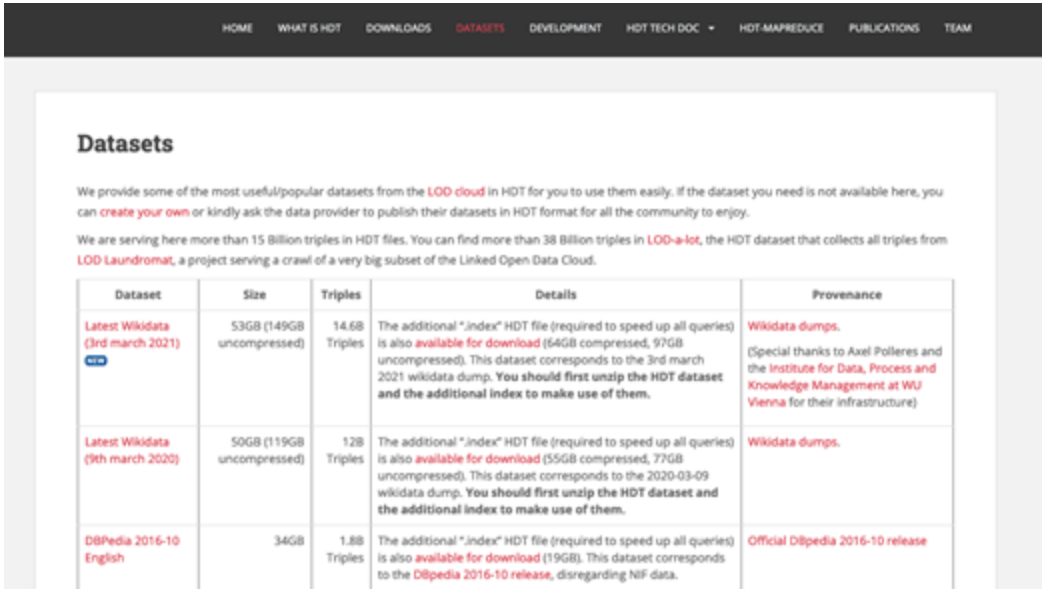
	Process	Type of Index	Patterns
1	index the bitsequences	Subject SPO	SPO, SP?, S??, S?O, ???
2	index the position of each predicate (just a position list)	Predicate PSO	?P?, ?PO
3	index the position of each object	Object OPS	??O

separate index file , created by consumer client (or published as as well)

Martínez-Prieto, M., M. Arias, and J. Fernández (2012). Exchange and Consumption of Huge RDF Data. In Proc. of the 9th Extended Semantic Web Conference (ESWC), pp. 437-452.

rdfhdt.org





https://github.com/rdfhdt C++ and Java tools

…. a bit like an "*ollama for KG querying*"

Outlook/Summary: HDT

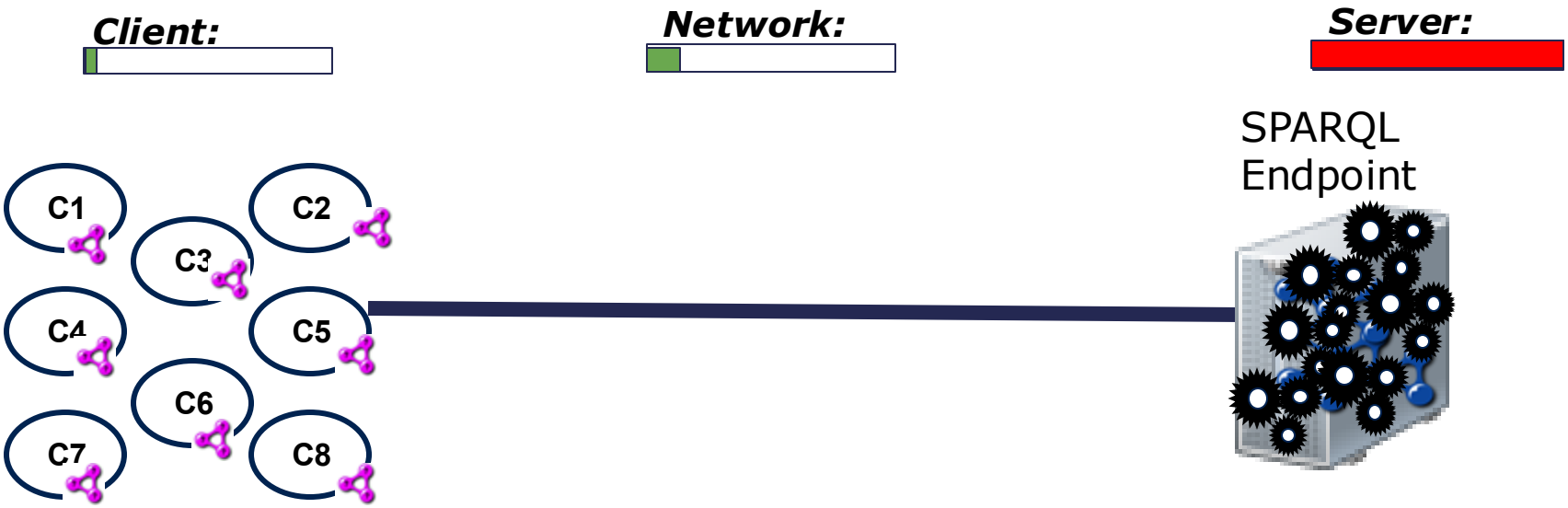
- Useful tool for **compressing, querying and exchanging large KGs (esp. triple patterns)**
- Data ready to be consumed in compressed format, 10-15x faster than loading it into an RDF triple store
  - HDT size << any other RDF format || RDF engine
- Competitive query performance.
  - Very fast on triple patterns, x 1.5 faster (Virtuoso, RDF-3x).
- Integration with Jena
  - Joins on the same scale of existing solutions (Virtuoso, RDF-3x).

- Status quo:
  - Some company takeup (e.g. QA company, data.world), but Open Source HDT Development recently less active
  - Current RDF Stores like Qlever use similar indexing ideas, https://dl.acm.org/doi/10.1145/3132847.3132921

Challenges:

- Many queries on DBPEdia's and Wikidata's SPARQL endpoint **time out**
- **What can we do about it?**
  - *Bespoke (Compressed) Indexing* ***(HDT)***
  - *Partitioning* ***(smart-KG)***

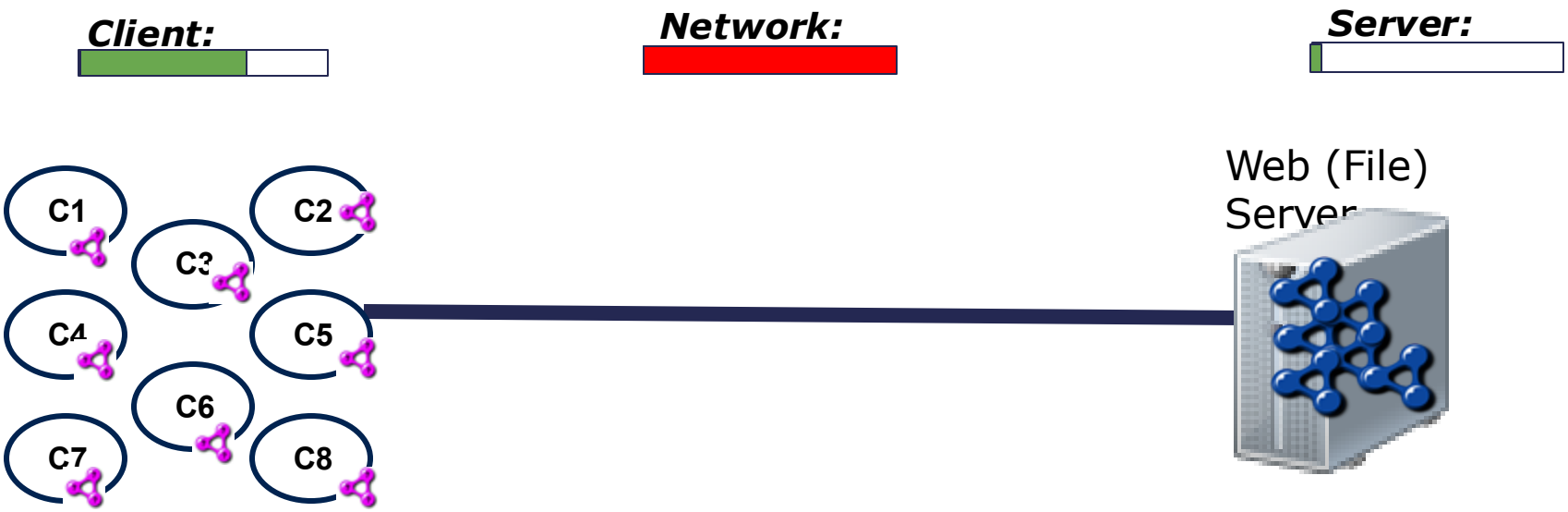
Server Solution: SPARQL Endpoint



"Query Shipping"

fails under concurrency

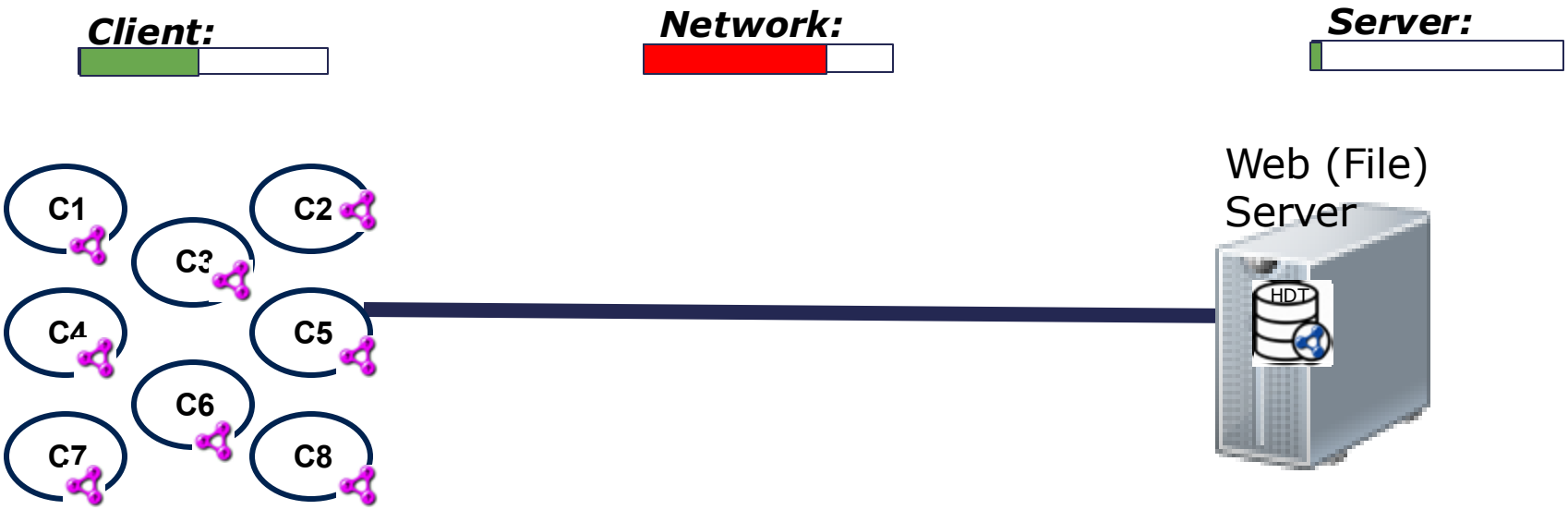
Client Solution: Data Dump



Data Shipping: Dumps

might add prohibitive load on the network

Client Solution: Compressed Dumps



Data Shipping – using HDT

still might add prohibitive load on the network
(e.g. DBpedia 2016 Dump 34GB for 1.8B triples)

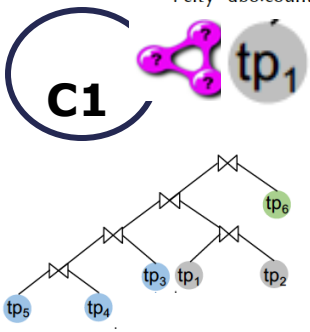
Triple Pattern Fragments (TPF):

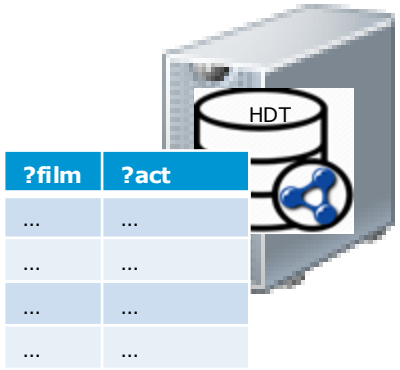
- Idea:
  - Execute single triple patterns on the server
  - Let the clients do JOINs etc. by themselves.

→ less footprint on the server, only triple patterns and intermediate results communicated.

→ can still have significant overhead by large intermediate results

```
SELECT * WHERE {
 ?film  dbo:starring  ?actress  .                  # tp1
 ?film  foaf:name ?filmName .                      # tp2
 ?actress  dbo:wikiPageExternalLink  ?link  .      # tp3
 ?actress  dbo:birthPlace  ?city  .                # tp4
 ?actress  foaf:gender "female"@en .               # tp5
 ?city  dbo:country  ?country  . }                 # tp6
```

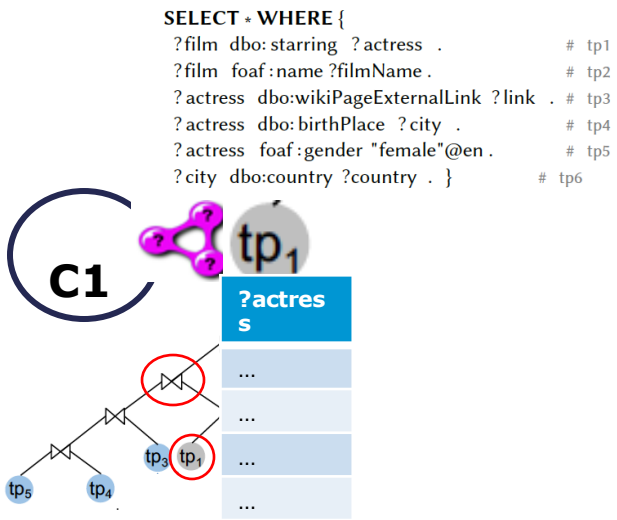


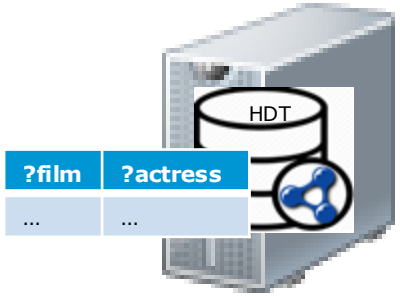


R. Verborgh,, M. van der Sande, O. Hartig, J. Van Herwegen, L. De Vocht, B. De Meester, G. Haesendonck, P. Colpaert: Triple Pattern Fragments: A low-cost knowledge graph interface for the Web. J. Web Semant. 37-38: 184-206 (2016)

Refinement: Binding-restricted Triple Pattern Fragments (br-TPF):

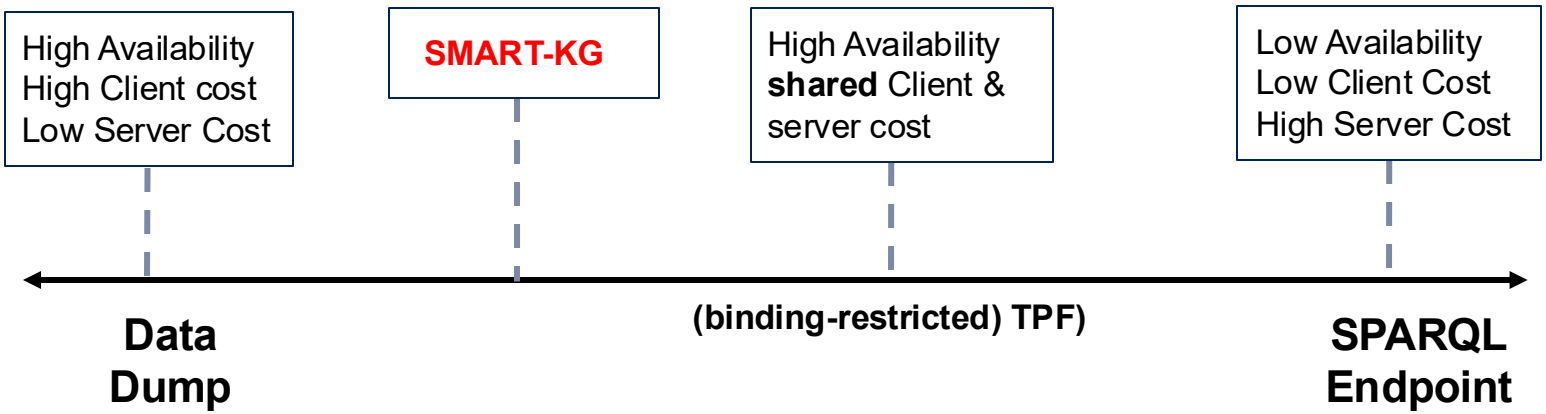
- Idea:
  - ship intermediate bindings with TP and let server only return results matching results
  - → smaller intermediate results, "join work" distributed between client and server





O. Hartig and C. B. Aranda. 2016. Bindings-Restricted Triple Pattern Fragments. In ODBASE 2016. 762–779

Can we do better? Remaining Problems:

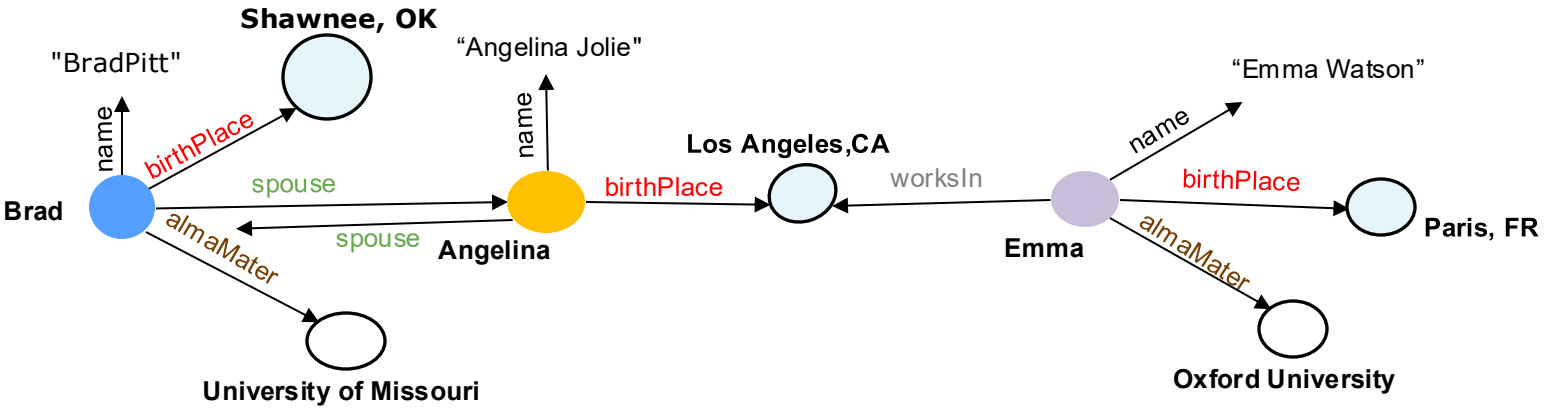


- Our experiments show that in highly concurrent query loads, with TPF:
  - still the server might get blocked
  - still an unncecessarily high number of (uncompressed) intermediate results may be shipped

Idea 1: "Partition" Shipping

smart-KG server: "Family" Partitions

Partition Generator (PG): Upon loading a graph KG G, decompose it into partitions $G_1,\ldots,G_m$, one per "predicate family".



F1: {name, birthPlace, spouse, almaMater}

F2: {name, birthPlace, spouse}

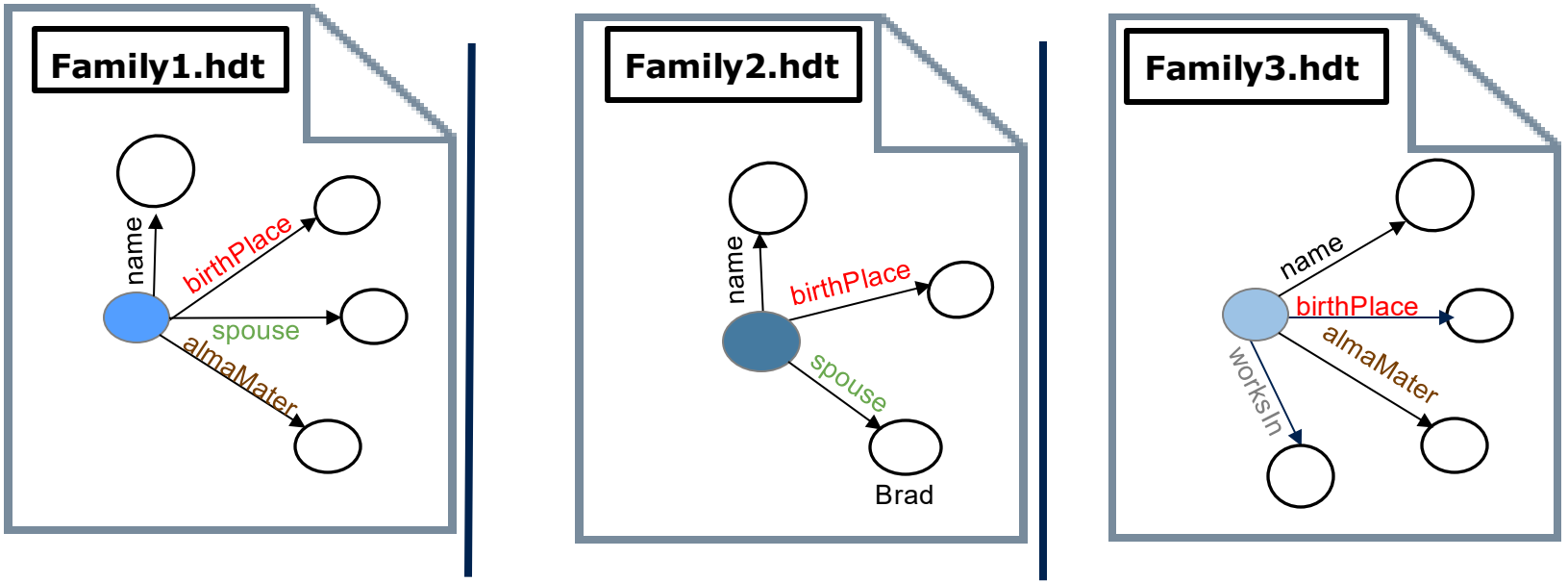
F3: {name, birthPlace,almaMater, worksIn}

F4: {name, country, capitalOf}

F4: {name, founded, numberOfStudents}

smart-KG server: Predicate "Family" Generator

Partition Generator (PG): Upon loading a graph KG G, decompose it into partitions $G_1,\ldots,G_m$, one per "predicate family" … and convert these to HDTs.



Smart-KG query Processing:

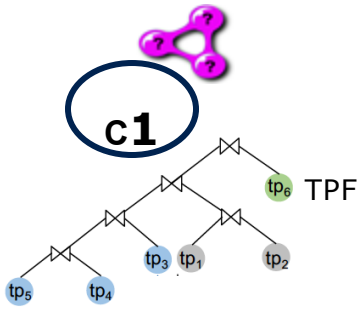
1. Client decomposes BGPs into "stars"
2. Retrieve relevant information from server to make a query plan
3. Retrieve and joins matching HDT partitions one by one

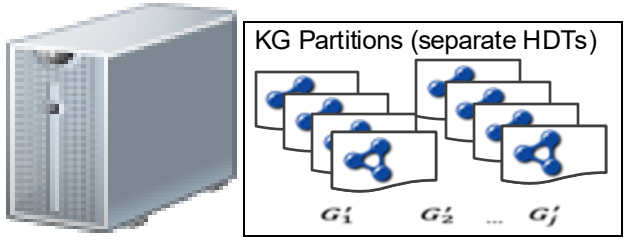
(use TPF for 1-triple patterns and "pruned" partitions)

```
SELECT * WHERE {
?film  dbo: starring  ? actress  .        # tp1
?film  foaf : name ?filmName .            # tp2
? actress  dbo:wikiPageExternalLink  ? link  .  # tp3
? actress  dbo: birthPlace  ? city  .     # tp4
? actress  foaf : gender  "female"@en .   # tp5
? city  dbo:country  ?country  . }        # tp6
```

F?:{starring, name}

F?:{wikiPageExtLink, birthPlace, gender}

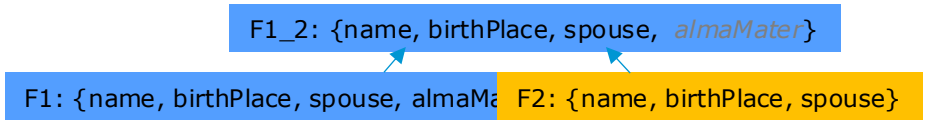




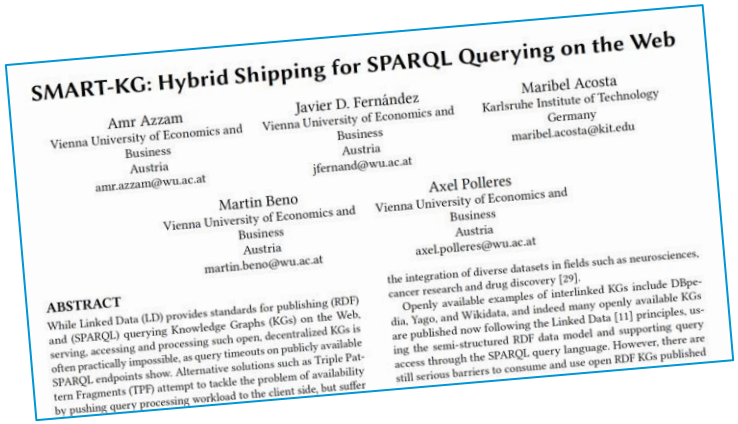
Smart-KG:

- Further details, cf. [8]:
  - predicate-restricted families, i.e. pruning+merging:
    - too rare or
    - too common

    predicates for partitioning e.g



Example: for **DBpedia**, a naive partitioning would create +600k partially very large families, which are unfeasible to serve.



SMART-KG: Hybrid Shipping for SPARQL Querying on the Web

Amr Azzam
Vienna University of Economics and Business
Austria
amr.azzam@wu.ac.at

Javier D. Fernández
Vienna University of Economics and Business
Austria
jfernand@wu.ac.at

Maribel Acosta
Karlsruhe Institute of Technology
Germany
maribel.acosta@kit.edu

Martin Beno
Vienna University of Economics and Business
Austria
martin.beno@wu.ac.at

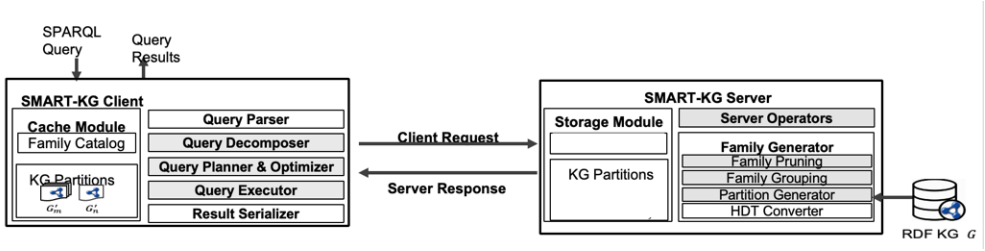
Axel Polleres
Vienna University of Economics and Business
Austria
axel.polleres@wu.ac.at

ABSTRACT

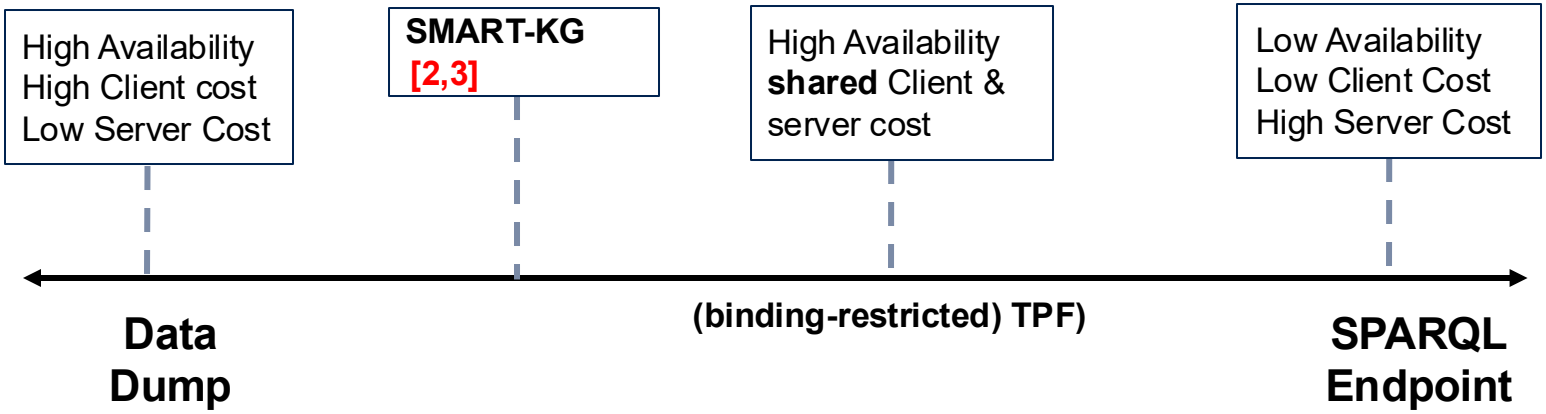
While Linked Data (LD) provides standards for publishing (RDF) and (SPARQL) querying Knowledge Graphs (KGs) on the Web, serving, accessing and processing such open, decentralized KGs is often practically impossible, as query timeouts on publicly available SPARQL endpoints show. Alternative solutions such as Triple Pattern Fragments (TPF) attempt to tackle the problem of availability by pushing query processing workload to the client side, but suffer

the integration of diverse datasets in fields such as neurosciences, cancer research and drug discovery [29].

Openly available examples of interlinked KGs include DBpedia, Yago, and Wikidata, and indeed many openly available KGs are published now following the Linked Data [11] principles, using the semi-structured RDF data model and supporting query access through the SPARQL query language. However, there are still serious barriers to consume and use open RDF KGs published



Can we do (even) better? Refinements:



- In partition shipping, the server is mainly a Web Server distributing partial dumps
- Desideratum: Server should process queries "as far as possible"

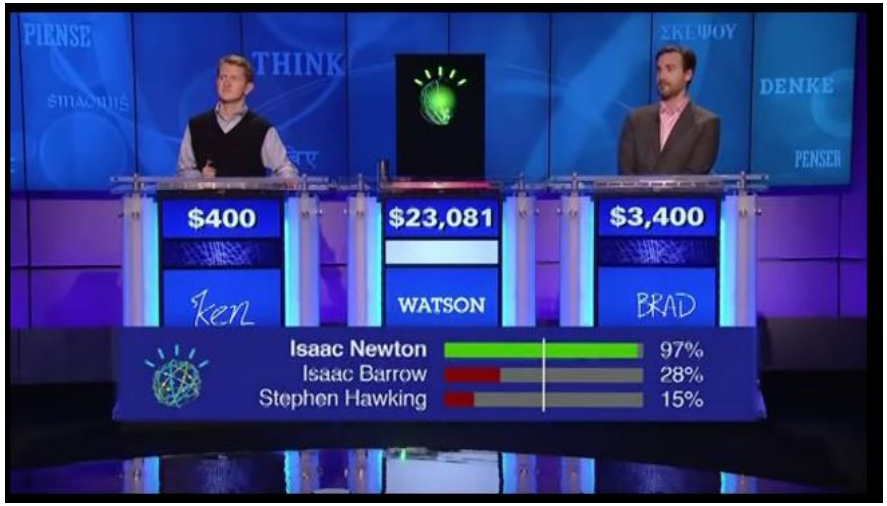
[3] Combined client & server processing
[2] Further refinement partitioning per node types (rather than predicates only).

Possible Discussion/Further ideas:

- Could similar ideas for modularization work for (agent-based) models?
- Combination of agent frameworks and KGs in extensions of (modular) Graph RAG?
- (How to) take context information into account for partitioning/indexing?

Another application:
How good or bad are KGs with Question answering?

- How good or bad are KGs with Question can answering?
- Interesting Note - IBM Watson - Jeopardy! (2011)
  "Super-human" Question Answering was achieved by Knowledge Graphs before the LLM hype!



DBpedia

Our own contribution in this area:

- ***Svitlana Vakulenko***, *Javier Fernández, Axel Polleres, Maarten de Rijke, and Michael Cochez. Message passing for complex question answering over knowledge graphs. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM2019, pages 1431--1440, Beijing, China, November 2019. ACM.*

will return to us via VRG W|W|T|F *grant!*

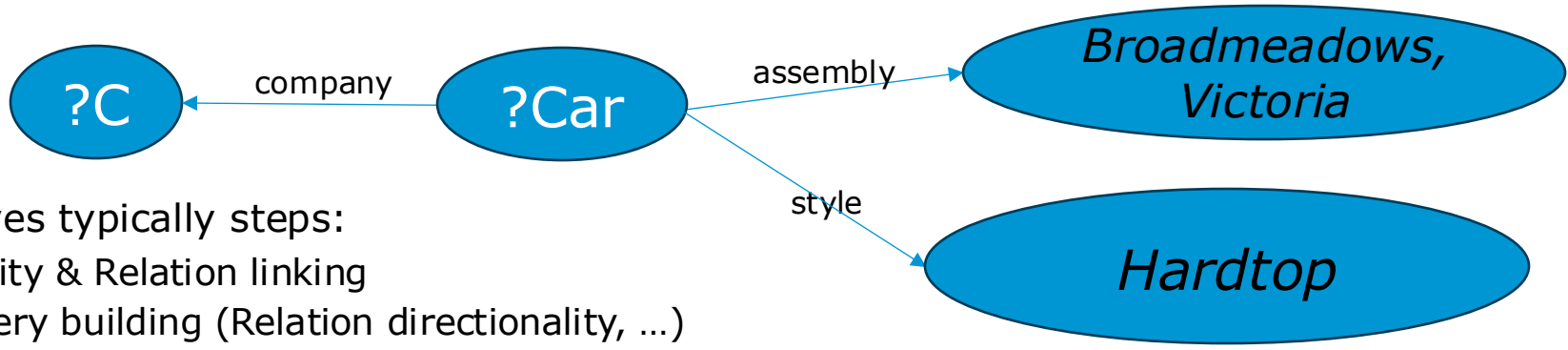
Another application of HDT: "Message-passing based" KGQA

Typical approach

- Natural Language question:

"*Which **company assembles** its **hardtop style** cars in **Broadmeadows, Victoria**?*"

- Map to a query (or graph matching) on a KG



- Involves typically steps:
  - Entity & Relation linking
  - Query building (Relation directionality, …)
  - …

```
SELECT ?C
WHERE {   ?Car dbo:company ?C .
          ?Car dbo:assembly .
          ?Car style dbr:Breadmedows.}
```

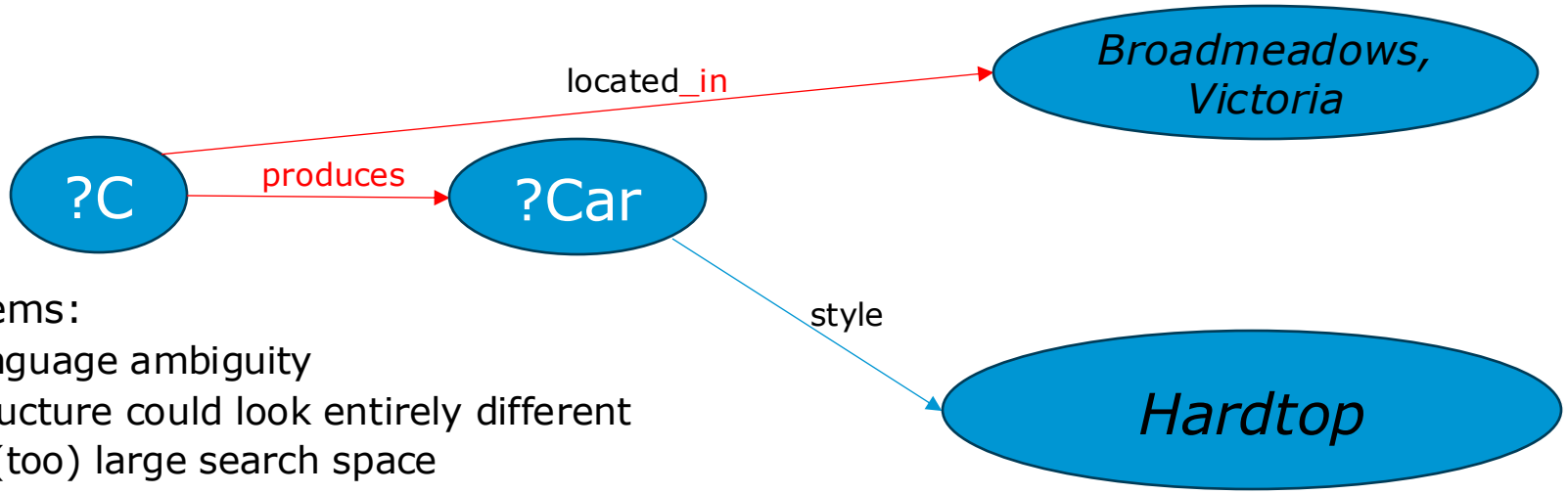
Another application of HDT: "Message-passing based" KGQA

Typical approach

- Natural Language question:

"*Which **company assembles** its **hardtop style** cars in **Broadmeadows, Victoria**?*"

- Map to a query (or graph matching) on a KG

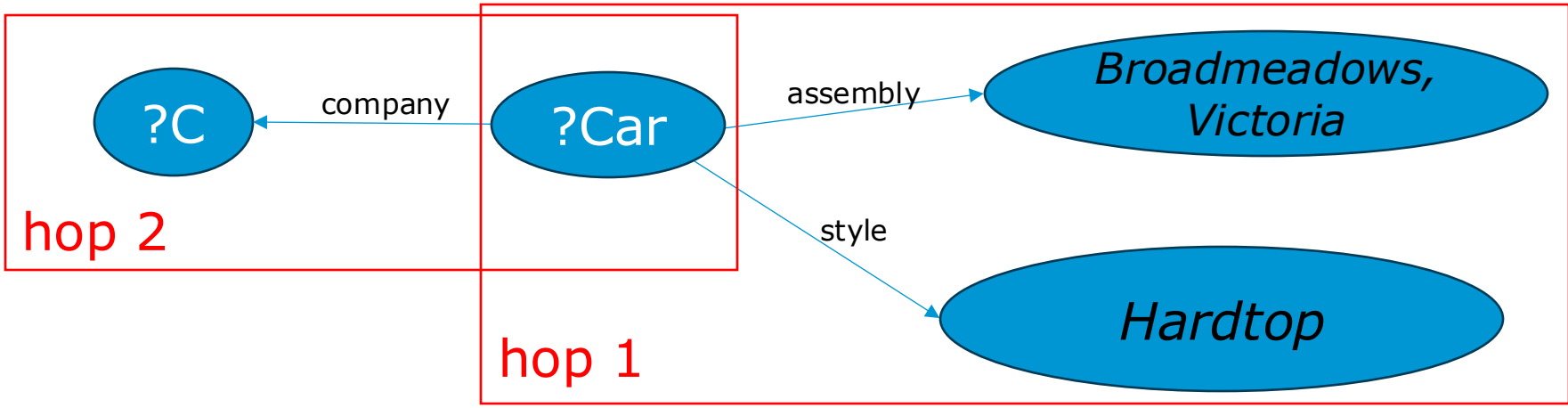


  - Problems:
    - Language ambiguity
    - Structure could look entirely different
    - → (too) large search space

Better approach:

- Step 1:
  - divide query into "hops" of simple (single "star-shaped") subquestions
  - Parse subquestion matching (in parallel) relevant entities and relations with ***confidence scores***
- Step 2*: propagating and aggregate these confidence scores over the KG via* ***message-passing***

→ *Idea: Possible answers are nodes with highest confidence*



We model questions (roughly) as sequences over a number of hops where each hop consists of

E … (candidate) entity sets,

P … candicate property sets,

C … candidate class sets the enity belong to

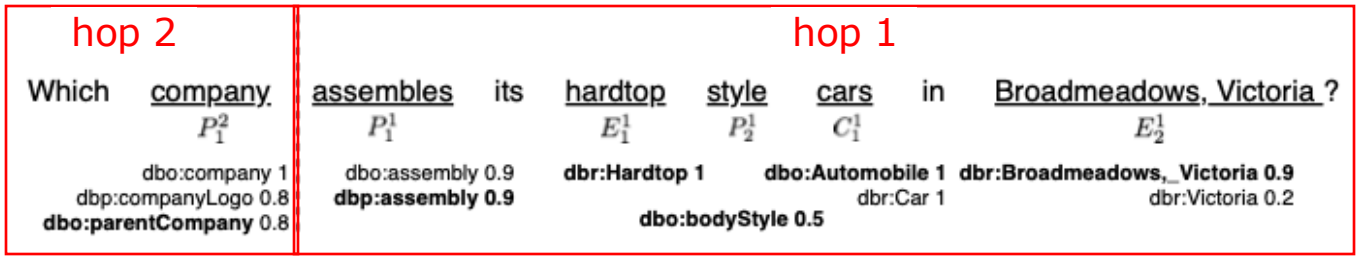
$$Seq_q = (\langle E^i, P^i, C^i \rangle)_{i=1}^{h}$$

Better appr

(a) Using sequence labeling with conditional random fields

(b) for each entity (or property, class, resp.) reference in, we retrieve a ranked list of most similar entities from the KG along with the matching confidence score.

- Step 1: (a) Parse question and (b) matching (in parallel) relevant entities and relations with ***confidence scores***

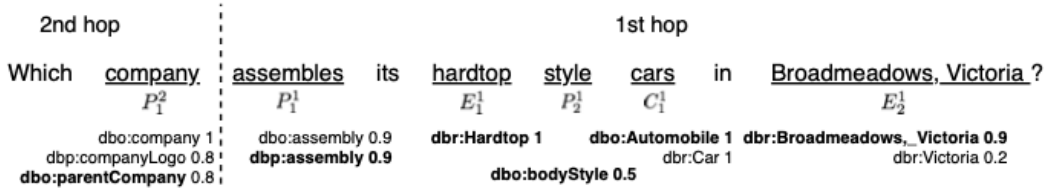


We model questions (roughly) as sequences over a number of hops where each hop consists of
E … (candidate) entity sets,
P … candicate property sets,
C … candidate class sets the enity belong to

$$Seq_q = (\langle E^i, P^i, C^i \rangle)_{i=1}^{h}$$

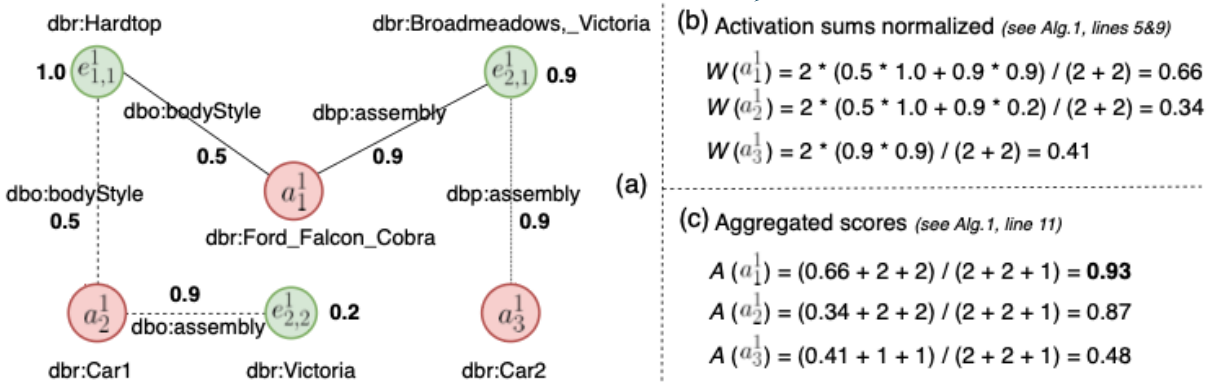
Better approach:

- Step 2*: propagating and aggregate these confidence scores over the KG via **message-passing (hop, by hop)***



- ***Hop 1 candidate subgraph and score aggregation:***

Note:
The weight propagation is less important than the fact that we use HDT for very efficient retrieval of the relevant subgraphs in each step in the implementation.
Note: once retrieved we do not consider direction of the edges in the original graph here.



a^i … answers in hop i

Bottomline:

- **message-passing** in the undirected subgraphs and
- **efficient retrieval** of these subgraphs (HDT), where sufficient to significantly improve over "query formulation" approaches on KGQA Benchmarks, e.g. QALD.
- Note/Disclaimer:
  - *That's where I left off*
  - *The part I contributed in the paper was minor.*

Discussion (we're only <u>starting</u> to look into this):  **+ ?**

- Could similar ideas improve other Graph tasks?
- How does this relate to other Graph Learning tasks that rely on efficient subgraph retrieval?
- Tying back to the first part: Do these tasks need the whole graph can they also benefit from modularization/partitioning?
- How scalable are other graph learning to Collaborative KGs?
- (How) do efficent representations for exact retrieval (i.e., indexing) relate to vectorized graph representations (embeddings)…?
- … and can we leverage retrieval-efficient representations in Graph-Learning?

Motivation

WU WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

In our group we have done a lot of work on

- Querying
- Analysing, and
- Improving the quality

of **Collaborative KGs at scale**

More things I did not talkabout:

Link analysis [5,6]

Constraint checking and repair [7, 

]

References:

1. Javier D. Fernández, Miguel A. Martinez-Prieto, Claudio Gutiérrez, Axel Polleres, and Mario Arias. Binary RDF Representation for Publication and Exchange (HDT). Journal of Web Semantics (JWS), 19(2), 2013 https://dl.acm.org/doi/10.1016/j.websem.2013.01.002
2. Amr Azzam, Axel Polleres, Javier D. Fernandez, and Maribel Acosta. smart-KG: Partition-based linked data fragments for querying knowledge graphs. Semantic Web -- Interoperability, Usability, Applicability (SWJ), 15(5):1791--1835, 2024. http://dx.doi.org/10.3233/SW-243571
3. Amr Azzam, Christian Aebeloe, Gabriela Montoya, Ilkcan Keles, Axel Polleres, and Katja Hose. WiseKG: Balanced Access to Web Knowledge Graphs. In Proceedings of the Web Conference 2021, pages 1422–--1434, Ljubljana, Slovenia, 2021. ACM / IW3C2. https://doi.org/10.1145/3442381.3449911
4. Svitlana Vakulenko, Javier Fernández, Axel Polleres, Maarten de Rijke, and Michael Cochez. Message passing for complex question answering over knowledge graphs. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM2019, pages 1431--1440, Beijing, China, November 2019. ACM. http://dx.doi.org/10.1145/3357384.3358026
5. Armin Haller, Javier D. Fernández, Maulik R. Kamdar, and Axel Polleres. What are links in linked open data? a characterization and evaluation of links between knowledge graphs on the web. *ACM Journal of Data and Information Quality (JDIQ)*, 2(2):1–--34, May 2020. http://dx.doi.org/10.1145/3369875
6. Romana Pernisch, Daniil Dobriy, and Axel Polleres. The massive problem of remote changes in ontology reuse. In *The Web Conference 2025*, Sydney, Australia, 2025. WWW25 Companion Proceedings, to appear (short paper)
7. Nicolas Ferranti, Jairo Francisco de Souza, Shqiponja Ahmetaj, and Axel Polleres. Formalizing and validating Wikidata's property constraints using SHACL and SPARQL. *Semantic Web -- Interoperability, Usability, Applicability (SWJ)*, 2024. https://journals.sagepub.com/doi/epub/10.3233/SW-243611
8. Shqiponja Ahmetaj, Iovka Boneva, Jan Hidders, Katja Hose, Maxime Jakubowski, Jose-Emilio Labra-Gayo, Wim Martens, Filip Murlak, Cem Okulmus, Axel Polleres, Ognjen Savković, and Mantas Simkus ans Dominik Tomaszuk. Common foundations for SHACL, PShEx, and PG-Schema. In *The Web Conference 2025*, Sydney, Australia, 2025. http://dx.doi.org/10.1145/3696410.3714694