On Enhancing Scalability for Distributed RDF/S Stores

George Tsatsanifos (NTUA-DBLAB)
Dimitris Sacharidis (IMIS R.C. "Athena")
Timos Sellis (NTUA, IMIS R.C. "Athena")

March 22, 2011

Outline

Introduction

Semantic Web Motivation

To describe web resources using formal language by creating metadata according to a formal representation of a discourse domain.

Resource Description Framework

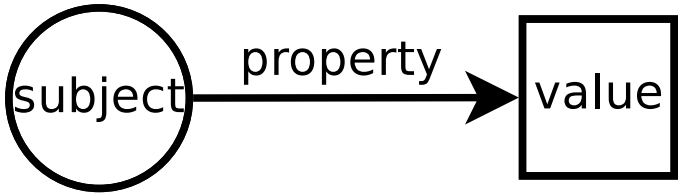
The RDF facilitates encoding, exchange, processing and reuse of resource metadata. Each community can specify its description semantics in a interoperable manner via an XML infrastructure.

The RDF Data Model

Directed Labeled Graphs

The constructs of the RDF data model describe interrelationships among resources in terms of named properties and values.

Nodes	Resources (URIs) or Literals
Edges	Properties, Attributes or Relationships
Labels	Nodes (Class names) and Edges (Property names)
Statement	Assertion of the form: `(resource, property, value)`
Description	Collection of statements concerning a resource



Constructs of the RDF/S meta-language

The RDF/S mechanism provides a type system for RDF models, a vocabulary of the valid terms that describe resources.

Abstraction via multiple class inheritance (`rds:subClassOf`, `rdf:subPropertyOf`) and multiple classification of resources (`rdf:type`).

Restriction declared on *domain* (`rdf:domain`) and *range* (`rdf:range`). The domain of a property consists of the classes having the declared property. The range consists of the values of the property.

Reification applicable at the data level (not in schema level) and the constructs used are *statement*, *subject*, *predicate*, *object* and *type*.

Querying the graph

SPARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage

SPARQL is a standard language to query RDF data.

```
SELECT ?x ?y WHERE {pattern1.  pattern2.  ··· }
```

"SPARQL will make a huge difference."

Sir Tim Burners-Lee, May 2006.

Distributed RDF Stores

Why?

- Increased flexibility
- Adhere to the open data model
- Sharing of knowledge
- Free access and dissemination of data and information
- Can scale to Internet size!

All of the above are indispensible parts of the Semantic Web vision.

RDFPeers

A structured P2P RDF store

- Leverages the MAAN overlay.
- Indexing a triple three time causes a replication factor of three.
- Extremely susceptible to load imbalances. In the DBLP dataset just seven predicates appear in the 72% of the triples.
- Queries with low selectivity require a linear number of hops in terms of the overlay size.

Different Approaches...

Many flavours!

1. There are solutions based on hierarchical approaches.
   - **Drago** based on ontologies.
   - **SONs** clusters of peers of similar content are formed.
   - **Marvin** combines clustering with random exchanges.
2. There are solutions based on unstructured overlays.
   - **Edutella** built over JXTA.
   - **Piazza** facilitates sharing of heterogeneous data.
   - **Chatty** relies on gossiping protocols

Inference Engines

In simple rule-based systems, there are two kinds of inference:

Forward chaining

Data-directed inference. Inference is triggered by the arrival of new data in working memory.

Backward chaining

Goal-directed inference, or hypothesis driven. Inferences are not performed till the system proves a particular goal (i.e. a question).

More complex methods use mixtures of forward and backward chaining.

Forward Chaining

1. Data gets put into working memory.
2. This triggers rules whose conditions match the new data.
3. These rules then perform their actions.
4. The actions add new data, triggering more rules.
5. And so on...

Labeling Schemes

Ideal for *efficiently* answering queries that require costly transitive closure computations.

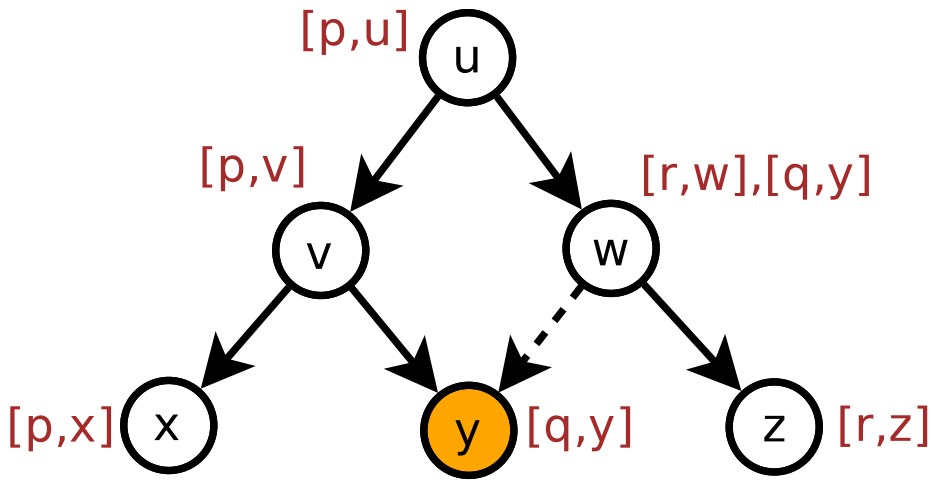
- Reachability
- Subsumption
- Descendants
- Nearest common ancestors

Interval Schemes

A range compression scheme for DAGs

1. Spanning tree T is determined.
2. A node u is labeled with $[\texttt{minpost}(u), \texttt{post}(u)]$.
3. Nodes are examined in reverse topological order.
4. For each edge (u, v) all intervals are propagated to u.
   - Subsumed intervals are discarded.
   - Overlapping intervals are merged.

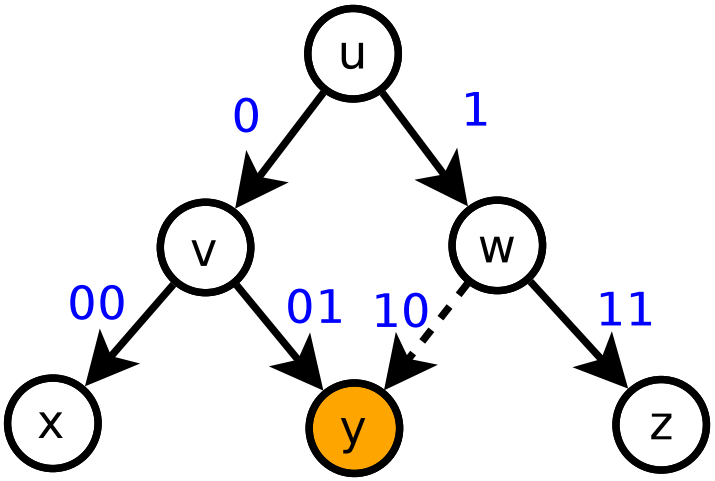
Strongly affected by the selection of the spanning tree.



The Dewey Prefix Scheme

Using a variable-length representation

- Consider alphabet $\Sigma = \{\sigma_1, \cdots, \sigma_M\}$.
- Node labels are defined recursively.
- Consider a node u with label $\texttt{id}(u) \in \Sigma$.
- We assign label $\texttt{id}(u)\sigma_k$ to the k-th child of u.



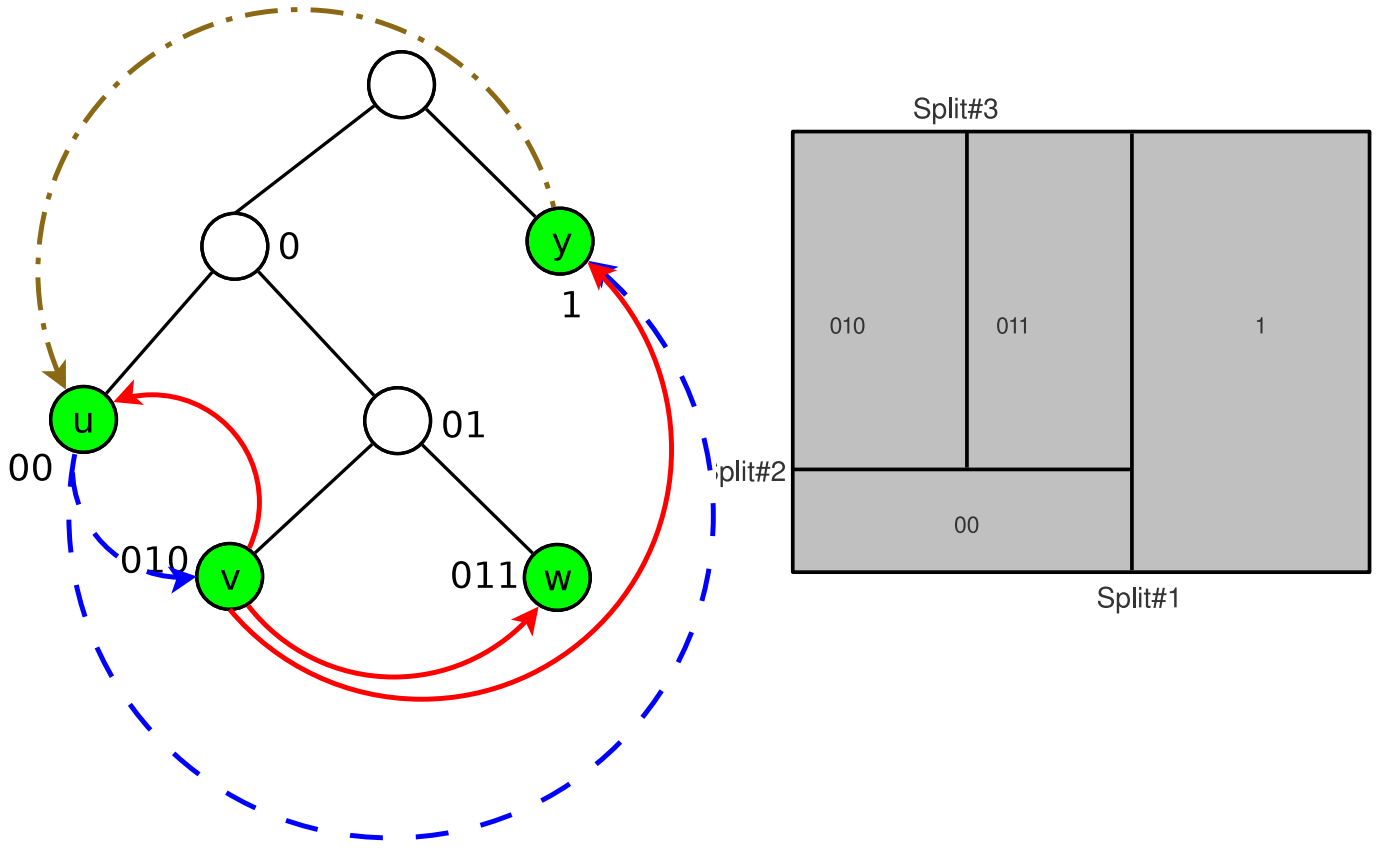
MIDAS under the scope

Multi-Attribute Indexing for Distributed Architecture Systems

- Virtual distributed kd-tree.
- Leaf nodes correspond to actual peers
- Internal nodes serve as routing directives.
- Virtual tree hierarchy is represented by *split history* (node's path) and *split points* in each peer.
- Their combination defines the position of a responsibility area in a coordinated zone (range partitioned network).
- A peer knows another peer from the subtree it does not belong to for each node in its path from the root.
- Tuples stored into the leaf nodes of the appropriate responsibility area.

Structure

Example



Storage Model

- Assume a lexicographic ordering for the subject, predicate, object, and the natural ordering for the labels.
- A RDF triple (u, α, v) is represented as a four dimensional key $\langle u, \alpha, v, id(v) \rangle$ when indexing.
- Field $id(v)$ encodes transitive information.
- For interval schemes, in a $(\vec{key}, value)$ pair, a composite value consists of all intervals associated with the key.

We are the first to endorse a mechanism that exploits labeling schemes in large-scale network applications!

Range Queries

An orthogonal search example

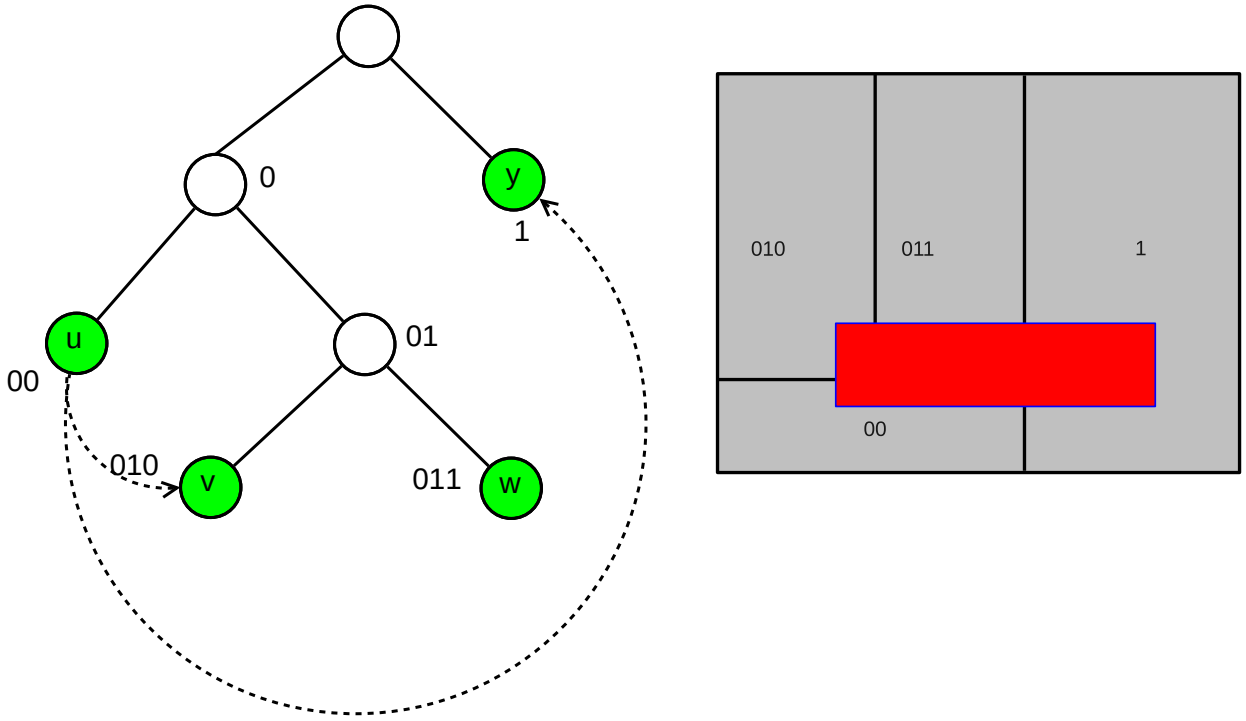


Figure: A range query example issued by 00 for the area depicted as red.

Range Queries

An orthogonal search example

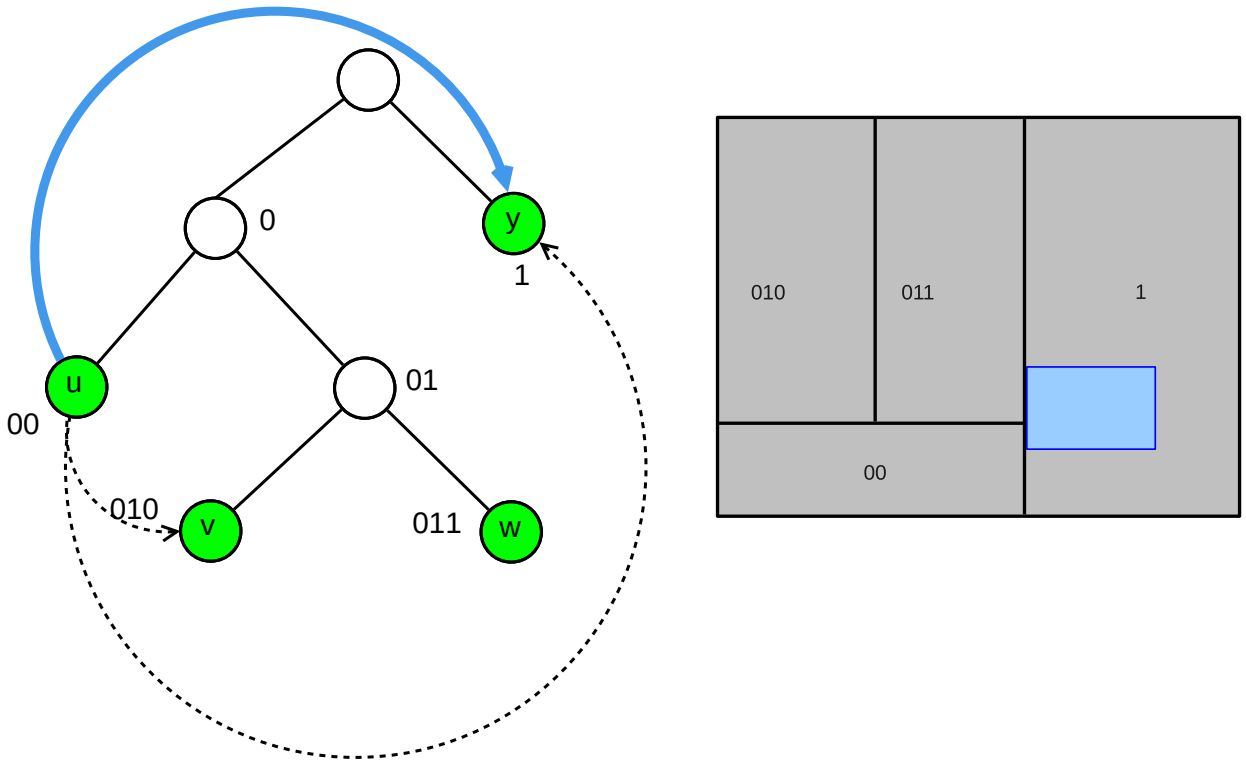


Figure: A range query example, partially answered by 1 (one hop) - query fragmented along the first split-point.

Range Queries

An orthogonal search example

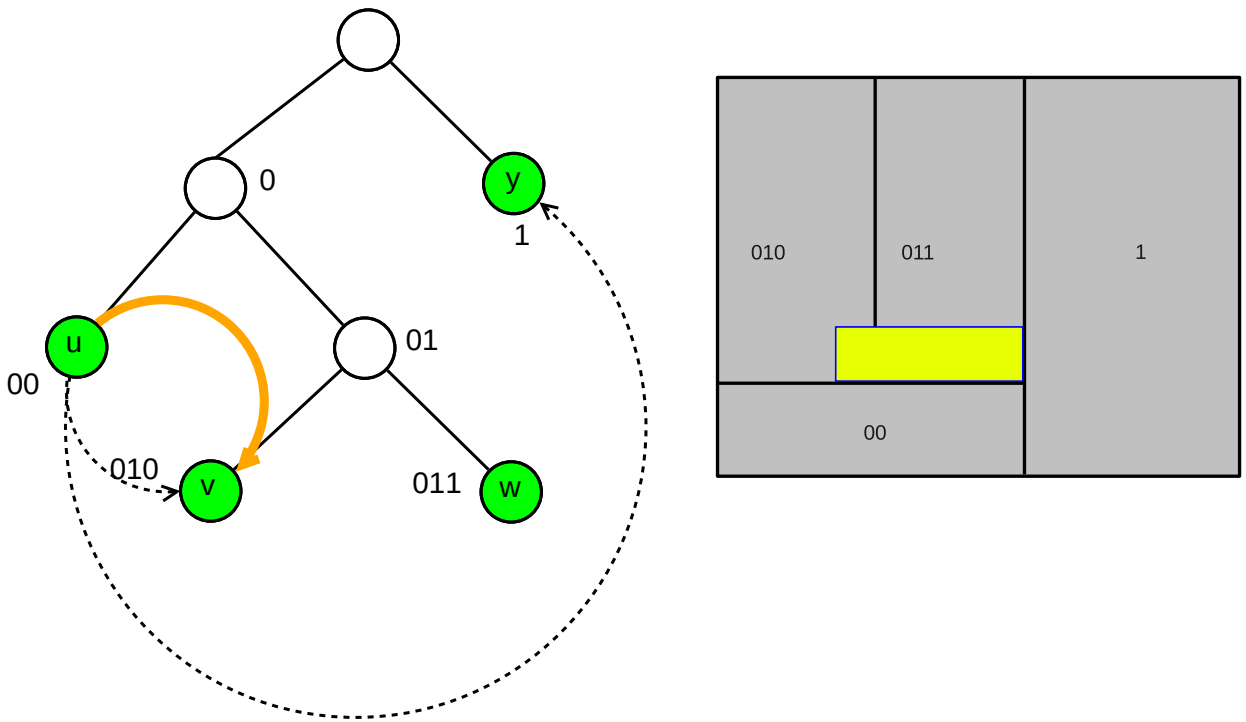


Figure: A range query example, partially answered by 010 (one hop) - subquery fragmented along a third split-point.

Range Queries

An orthogonal search example

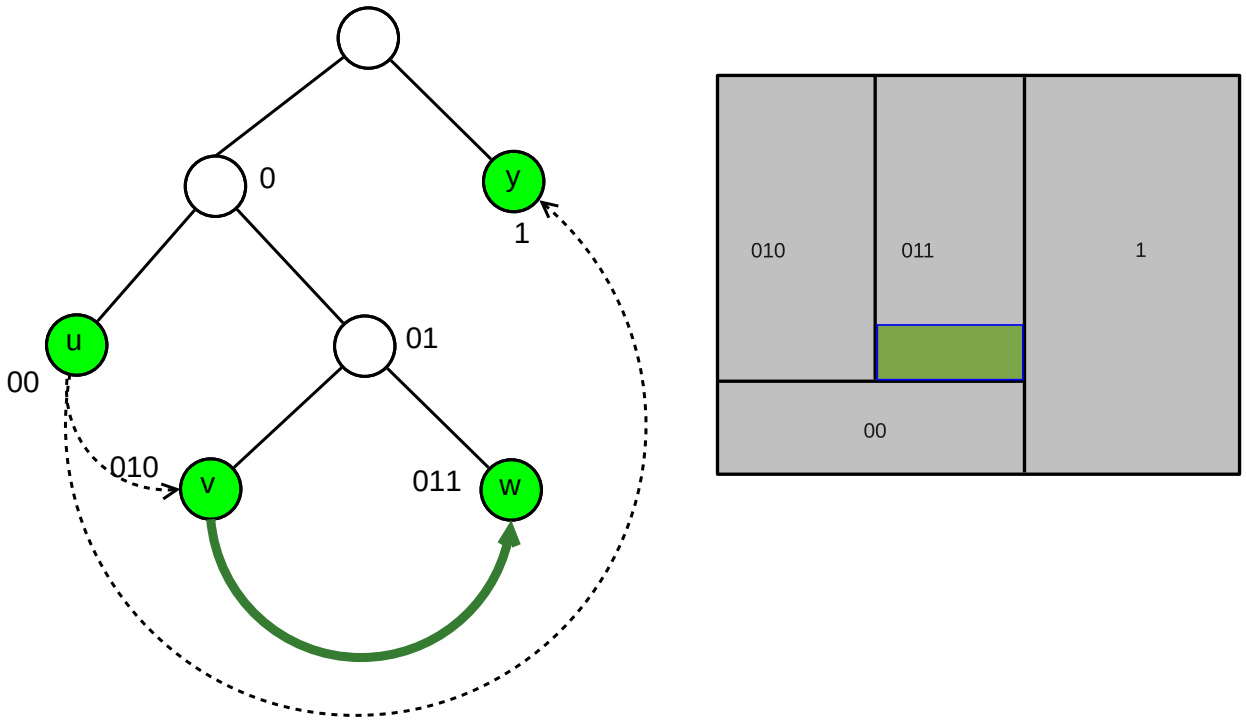


Figure: A range query example, partially answered by 011 (two hops).

Range Queries

An orthogonal search example

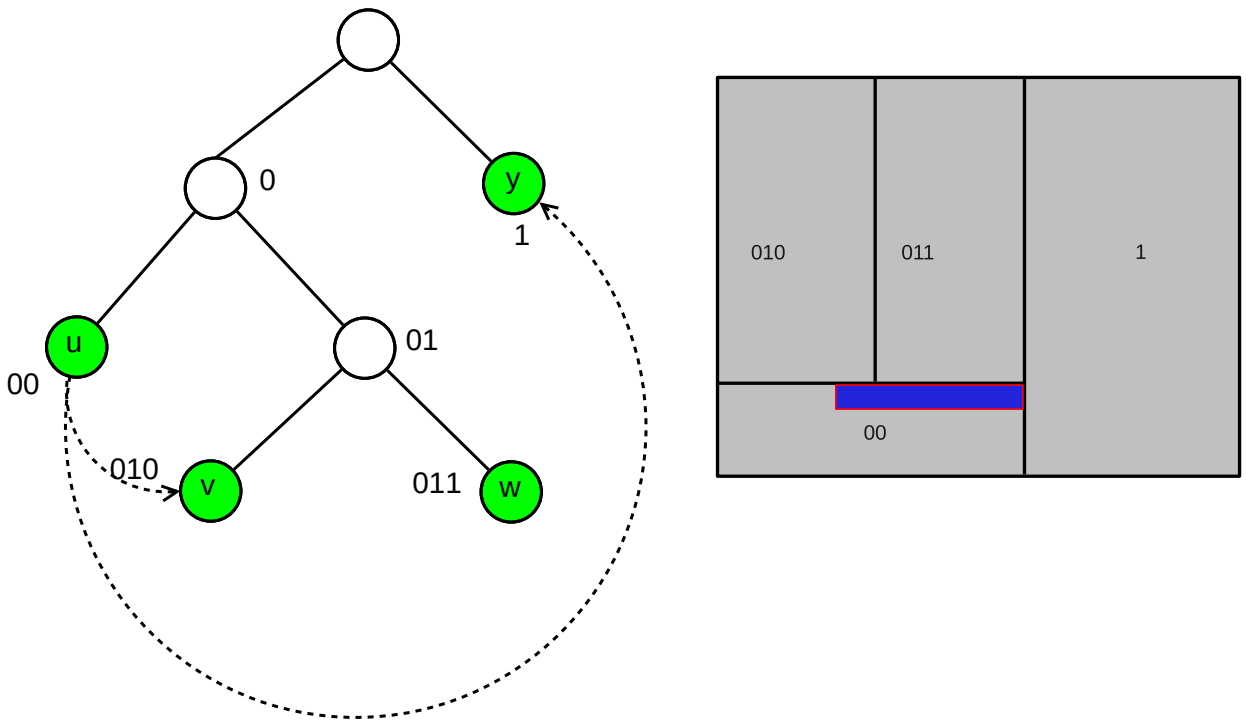


Figure: A range query example, partially processed locally by the issuing node 00 - subquery fragmented along a second split-point.

Range Queries

An orthogonal search example

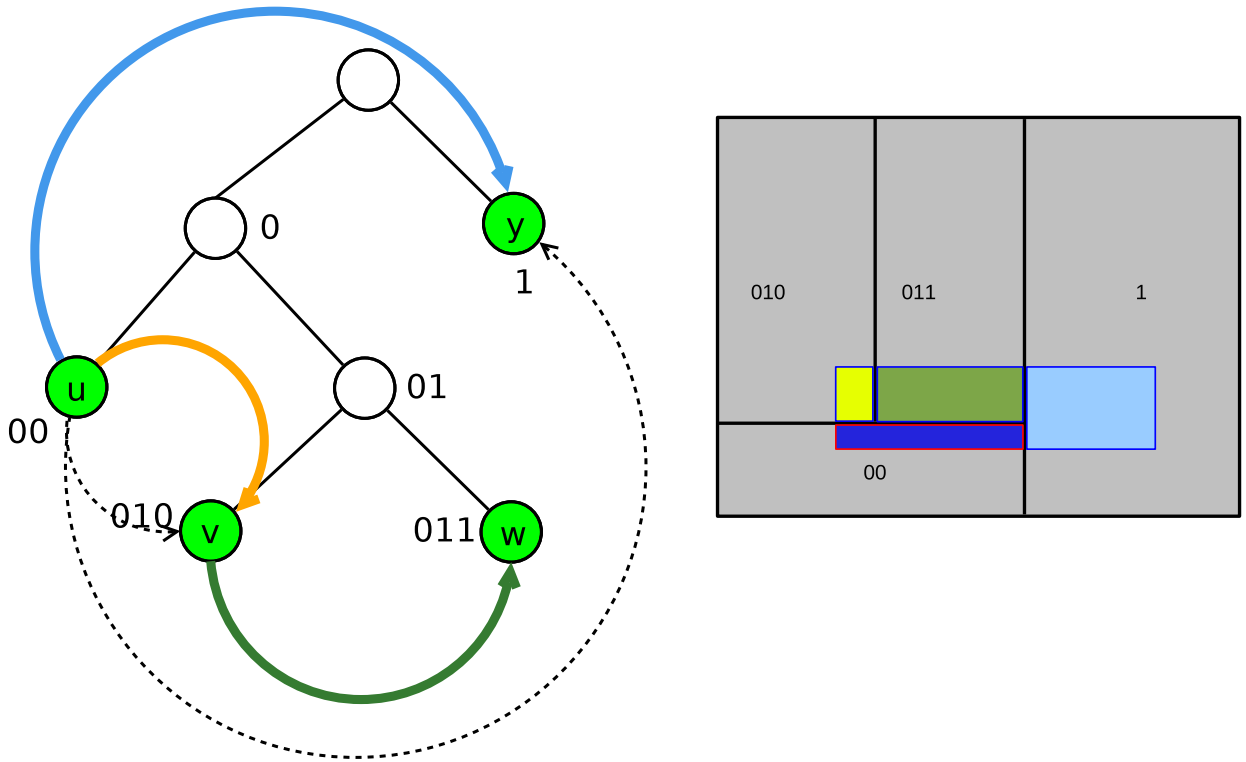


Figure: A range query example, issued by 00 and partially answered by 1 and 010 (one hop), and 011 (two hops).

Atomic Queries

Queries Q2-Q8 are transformed into range queries!

Name	Subject	Predicate	Object
Q1	s	p	o
Q2	s	p	?o
Q3	s	?p	o
Q4	s	?p	?o
Q5	?s	p	o
Q6	?s	p	?o
Q7	?s	?p	o
Q8	?s	?p	?o

RDF Queries

Disjunctive Queries

Issue one query for each contiguous range and then compute the union of the results.

Conjunctive Queries

Invoke first the atomic query that constitutes the conjunction with greater selectivity, and then, filter the result-set appropriately.

Transitive Closure Computation

for interval schemes

Retrieve the triples with entities subsumed by resource u having the property α (e.g., `subClassOf`).

Interval Based Scheme

1. Query for the peer responsible for u.
2. Retrieve the associated value.
3. Parse the value into the intervals with the identifiers of the subsumed resources.
4. Issue simultaneously range queries for all of them.

Resolves in $O(\log n)$ hops, regardless the depth of the relation!

Transitive Closure Computation

for prefix schemes

Retrieve the triples with entities subsumed by resource u having the property α (e.g., `subClassOf`).

Prefix Based Scheme

1. Query for the peer responsible for u.
2. Retrieve the label field $\texttt{id}(u)$.
3. $\texttt{id}(u)$ is the prefix of the identifiers of all subsumed nodes.
4. Issue a range query for the tuples labeled in $[\texttt{id}(u), \texttt{id}(w))$.

Node w is u's next sibling if it exists; otherwise, it is the label of the next sibling of its parent, and so on.

W3C RDFS Entailment Rules

Rule	Precondition	Generated Triple
rdfs2	$(\alpha, \texttt{domain}, x), (u, \alpha, v)$	$(u, \texttt{type}, x)$
rdfs3	$(\alpha, \texttt{range}, x), (u, \alpha, v)$	$(v, \texttt{type}, x)$
rdfs5	$(\alpha, \texttt{sp}, \beta), (\beta, \texttt{sp}, \gamma)$	$(\alpha, \texttt{sp}, \gamma)$
rdfs7	$(\alpha, \texttt{sp}, \beta), (u, \alpha, v)$	(u, β, v)
rdfs9	$(u, \texttt{sc}, v), (w, \texttt{type}, u)$	$(w, \texttt{type}, v)$
rdfs11	$(u, \texttt{sc}, v), (v, \texttt{sc}, w)$	$(u, \texttt{sc}, w)$
ext1	$(\alpha, \texttt{domain}, u), (u, \texttt{sc}, v)$	$(\alpha, \texttt{domain}, v)$
ext2	$(\alpha, \texttt{range}, u), (u, \texttt{sc}, v)$	$(\alpha, \texttt{range}, v)$
ext3	$(\alpha, \texttt{domain}, u), (\beta, \texttt{sp}, \alpha)$	$(\beta, \texttt{domain}, u)$
ext4	$(\alpha, \texttt{range}, u), (\beta, \texttt{sp}, \alpha)$	$(\beta, \texttt{range}, u)$

With the exception of rules RDFS2 and RDFS3, all others include transitive relations!

Applying Entailment Rule RDFS5

MIDAS-RDF - READY? GO!

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	
p_2	$(\beta, \mathtt{sp}, \zeta)$	
p_3	$(\zeta, \mathtt{sp}, \xi)$	
p_4	$(\xi, \mathtt{sp}, \psi)$	

Competitor - READY? GO!

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	
p_2	$(\beta, \mathtt{sp}, \zeta)$	
p_3	$(\zeta, \mathtt{sp}, \xi)$	
p_4	$(\xi, \mathtt{sp}, \psi)$	

Applying Entailment Rule RDFS5

MIDAS-RDF - DONE!!!

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	$(\alpha, \mathtt{sp}, \zeta), (\alpha, \mathtt{sp}, \xi), (\alpha, \mathtt{sp}, \psi)$
p_2	$(\beta, \mathtt{sp}, \zeta)$	$(\beta, \mathtt{sp}, \xi), (\beta, \mathtt{sp}, \psi)$
p_3	$(\zeta, \mathtt{sp}, \xi)$	$(\zeta, \mathtt{sp}, \psi)$
p_4	$(\xi, \mathtt{sp}, \psi)$	

Competitor - WORK IN PROGRESS...

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	$(\alpha, \mathtt{sp}, \zeta)$
p_2	$(\beta, \mathtt{sp}, \zeta)$	$(\beta, \mathtt{sp}, \xi)$
p_3	$(\zeta, \mathtt{sp}, \xi)$	$(\zeta, \mathtt{sp}, \psi)$
p_4	$(\xi, \mathtt{sp}, \psi)$	

Applying Entailment Rule RDFS5

MIDAS-RDF - OK, I'm waiting!

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	$(\alpha, \mathtt{sp}, \zeta), (\alpha, \mathtt{sp}, \xi), (\alpha, \mathtt{sp}, \psi)$
p_2	$(\beta, \mathtt{sp}, \zeta)$	$(\beta, \mathtt{sp}, \xi), (\beta, \mathtt{sp}, \psi)$
p_3	$(\zeta, \mathtt{sp}, \xi)$	$(\zeta, \mathtt{sp}, \psi)$
p_4	$(\xi, \mathtt{sp}, \psi)$	

Competitor - WORK IN PROGRESS...

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	$(\alpha, \mathtt{sp}, \zeta), (\alpha, \mathtt{sp}, \xi)$
p_2	$(\beta, \mathtt{sp}, \zeta)$	$(\beta, \mathtt{sp}, \xi), (\beta, \mathtt{sp}, \psi)$
p_3	$(\zeta, \mathtt{sp}, \xi)$	$(\zeta, \mathtt{sp}, \psi)$
p_4	$(\xi, \mathtt{sp}, \psi)$	

Applying Entailment Rule RDFS5

MIDAS-RDF

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	$(\alpha, \mathtt{sp}, \zeta), (\alpha, \mathtt{sp}, \xi), (\alpha, \mathtt{sp}, \psi)$
p_2	$(\beta, \mathtt{sp}, \zeta)$	$(\beta, \mathtt{sp}, \xi), (\beta, \mathtt{sp}, \psi)$
p_3	$(\zeta, \mathtt{sp}, \xi)$	$(\zeta, \mathtt{sp}, \psi)$
p_4	$(\xi, \mathtt{sp}, \psi)$	

Competitor - DONE!

Peer	Local Store	Generated Triples
p_1	$(\alpha, \mathtt{sp}, \beta)$	$(\alpha, \mathtt{sp}, \zeta), (\alpha, \mathtt{sp}, \xi), (\alpha, \mathtt{sp}, \psi)$
p_2	$(\beta, \mathtt{sp}, \zeta)$	$(\beta, \mathtt{sp}, \xi), (\beta, \mathtt{sp}, \psi)$
p_3	$(\zeta, \mathtt{sp}, \xi)$	$(\zeta, \mathtt{sp}, \psi)$
p_4	$(\xi, \mathtt{sp}, \psi)$	

Experimental Evaluation

Setting

- Our experiments simulate a dynamic environment.
- We initiate an overlay of 1K peers, increasing to 100K peers, followed by the reverse procedure.
- We used the Proximity DBLP dataset (11.2M triples).
- Synthetic graphs of variable depth, up to 16.
- Querysets consist of 40K queries.
- Important metrics: Latency, Congestion, Data Load.

DBLP Dataset Characteristics

A case study on predicates

Frequency	Relationships (in main relationships)
900,440	publication-has-author (author)
438,531	contained in proceedings (isIncludedIn)
112,303	cites publication
10,639	has-homepage (foaf:homepage)
10,461	has-publisher (dc:publisher)
7,308	has affiliation (foaf:workplaceHomepage)
5,850	in series

Table: Statistics summary of the DBLP dataset (Resource-to-Resource Triples: 3,740,438, Resource-to-Literal Triples: 7,274,180).

Results

for the DBLP workload

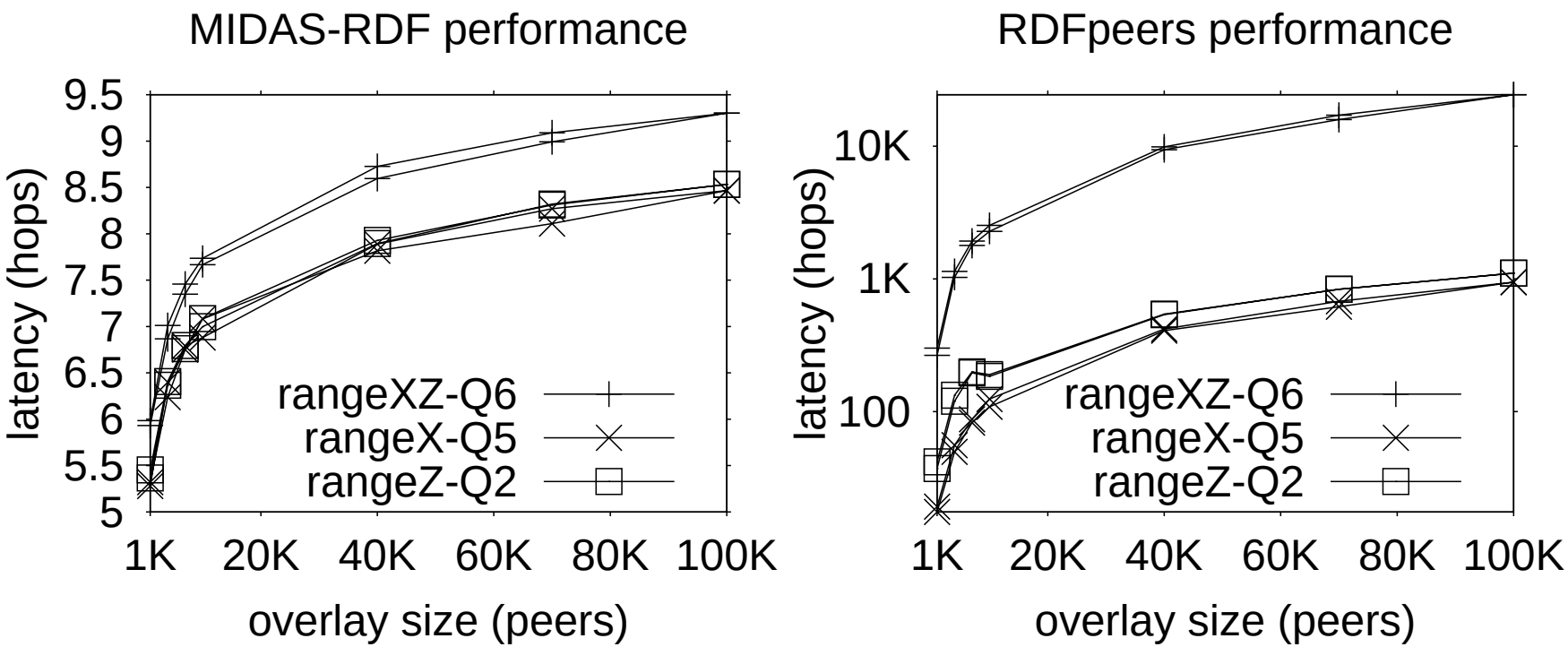


Figure: Latency of MIDAS-RDF, RDFPeers for *rangeXZ*, *rangeX*, *rangeZ* querysets on the DBLP dataset.

Results

for the synthetic graphs

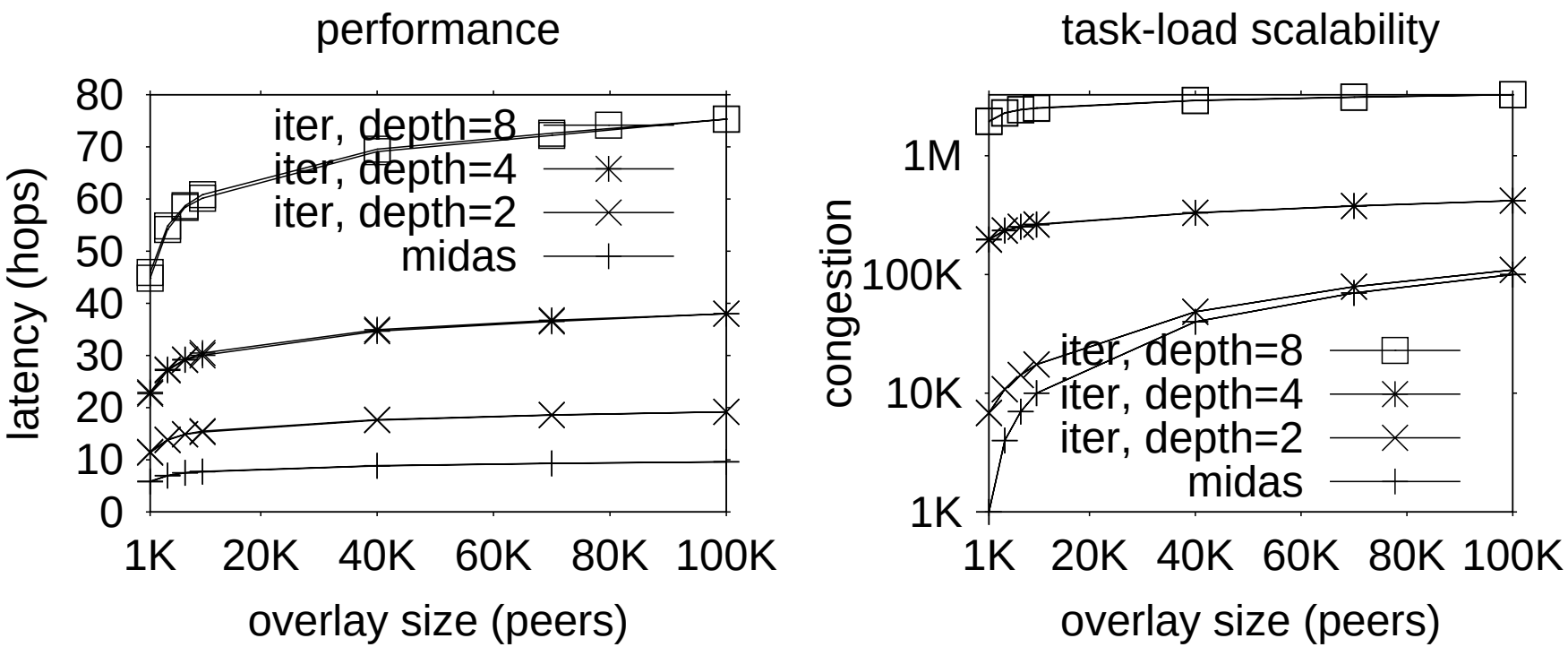


Figure: Latency and task-load for the computation of the deductive closure for labeling and iterative based schemes.

Conclusions

To recapitulate...

MIDAS-RDF is a novel distributed RDF/S repository and an inference engine that ...

1. relies on a pure multi-dimensional indexing scheme for large-scale decentralized networks.
2. consolidates labeling schemes as a means for enhancing performance and scalability.
3. latency shows logarithmic behavior to the overlay size.
4. outperforms any previous approach.
5. advances the state of the art.

Questions?