Decentralized Indexing over a Network of RDF Peers

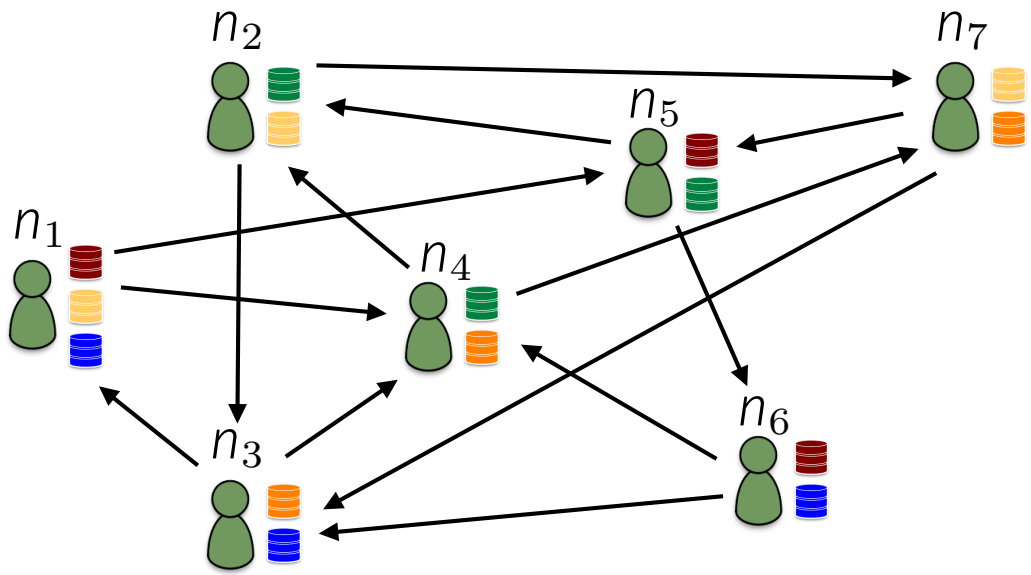
Christian Aebeloe Gabriela Montoya Katja Hose

Aalborg University, Denmark
{caebel,gmontoya,khose}@cs.aau.dk

SPARQL endpoints are *often unavailable*[1]

[1] C. Buil-Aranda et al. **SPARQL Web Querying Infrastructure: Ready for Action?** ISWC 2013

PIQNIC: A DECENTRALIZED ARCHITECTURE FOR SHARING AND QUERYING SEMANTIC DATA



Node

A node is $n = \langle \mathcal{G}_n, N, u \rangle$ where $\mathcal{G}_n$ is the set of graphs at n, N is n's neighbors, u is n's identifier

Datasets are *fragmented* based on predicate

C. Aebeloe et al. **A Decentralized Architecture for Sharing and Querying Semantic Data** ESWC 2019

QUERY PROCESSING IN PIQNIC

General algorithm

1. Reorder triple patterns based on *selectivity*

QUERY PROCESSING IN PIQNIC

General algorithm

1. Reorder triple patterns based on *selectivity*
2. Evaluate each triple pattern *tp*
    - Source *selection*
    - *Retrieval* of results from sources

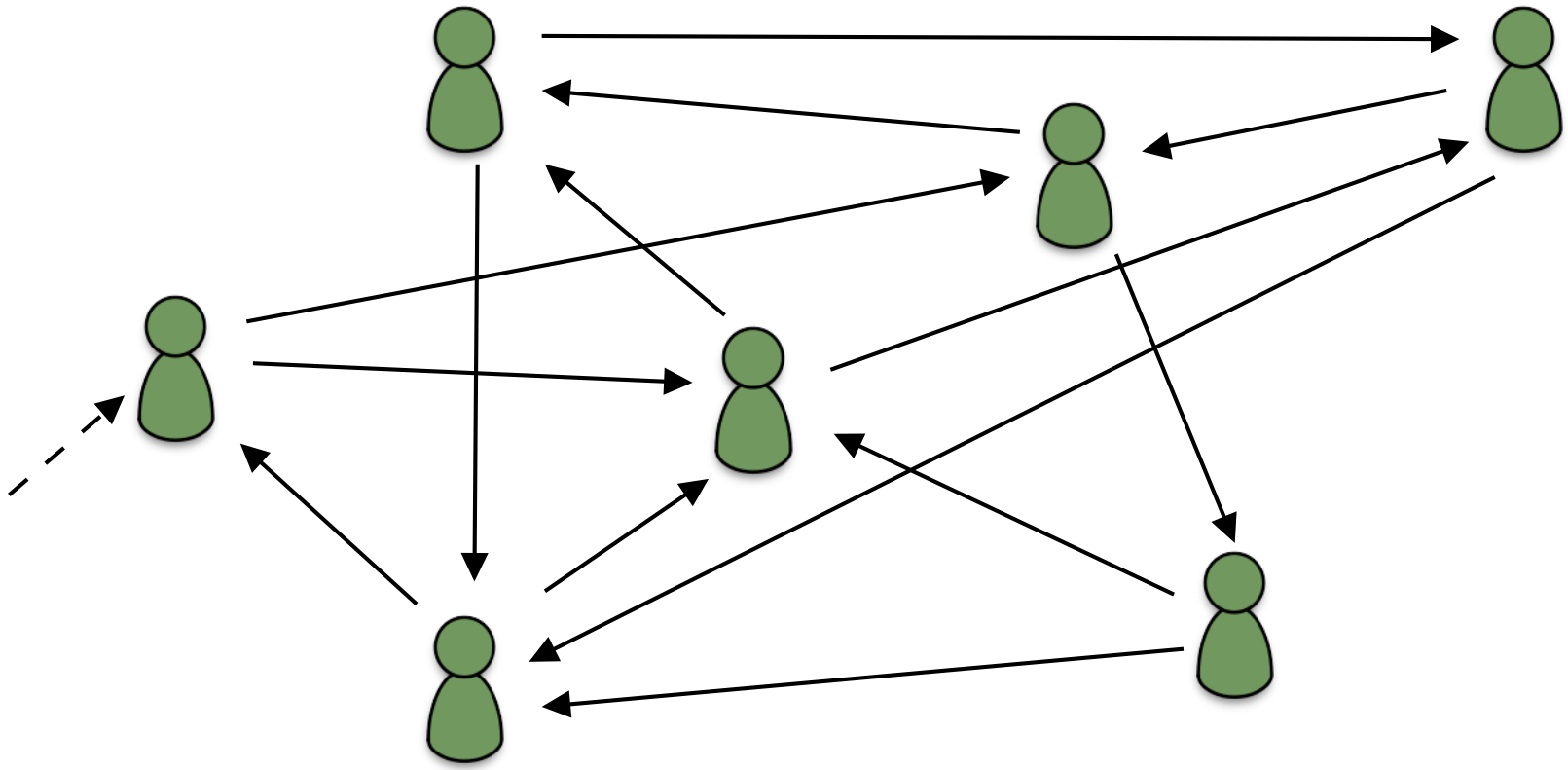
QUERY PROCESSING IN PIQNIC

General algorithm

1. Reorder triple patterns based on *selectivity*
2. Evaluate each triple pattern *tp*
   - Source *selection*
   - *Retrieval* of results from sources
3. Compute the full answer combining *previously obtained* bindings

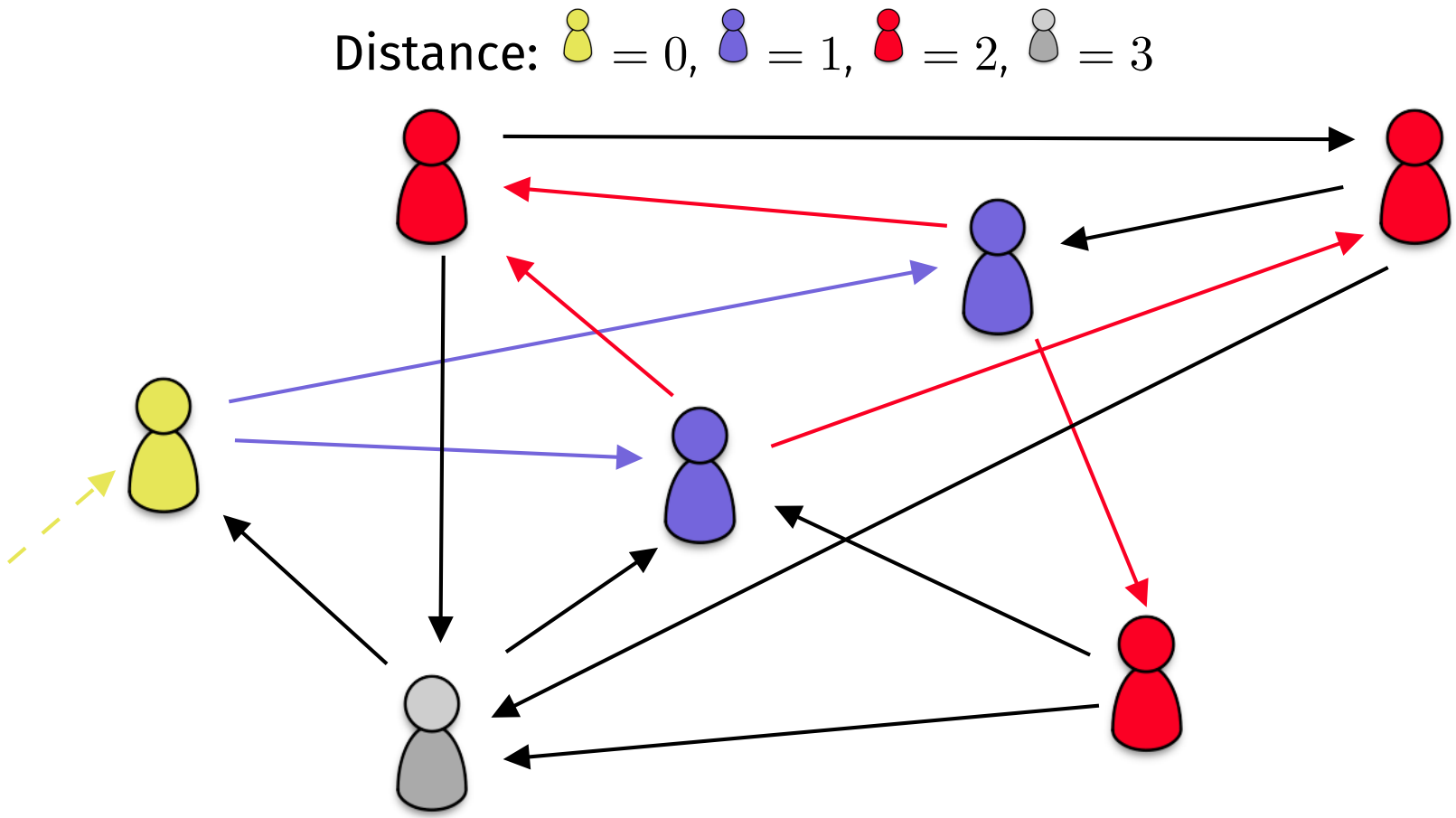
DECENTRALIZED QUERY PROCESSING

Basic strategy relies on *flooding*



DECENTRALIZED QUERY PROCESSING

Basic strategy relies on *flooding*



Number of *requests*: $\sum_{i=1}^{ttl} N^i$

LOCATIONAL INDEXES

Identify *relevant* nodes to ask

Locational Index

Let $\mathcal{N}$ be the set of nodes, $\mathcal{P}$ predicates, and $\mathcal{G}$ graphs. A locational index of depth i is $l^i_L(n) = \langle \gamma, \eta \rangle$ s.t. $\gamma : \mathcal{P} \rightarrow 2^{\mathcal{G}}$ and $\eta : \mathcal{G} \rightarrow 2^{\mathcal{N}}$

Given a *predicate*, identify relevant *graphs/fragments*

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

$\Rightarrow$

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

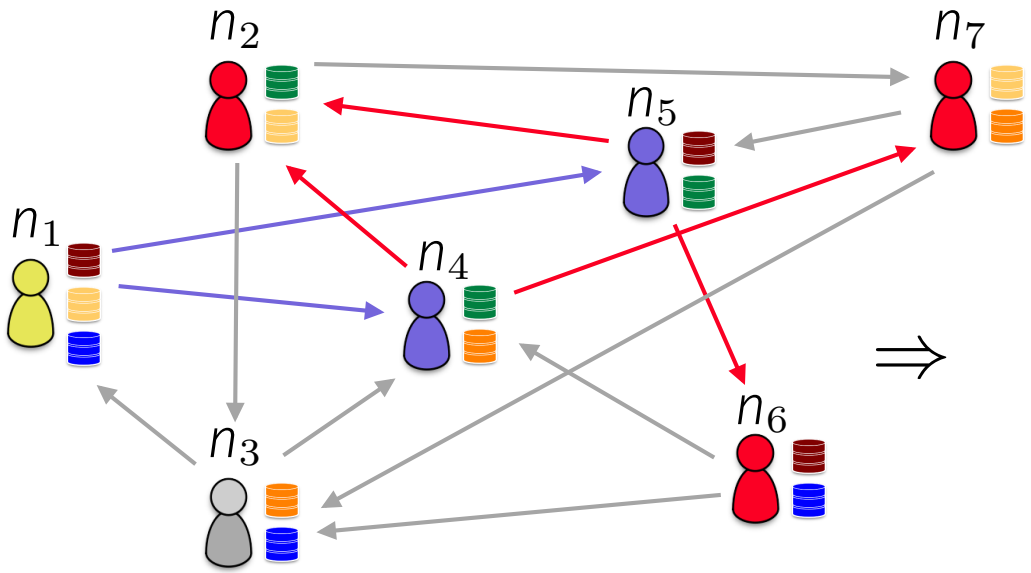
LOCATIONAL INDEXES

Identify *relevant* nodes to ask

Locational Index

Let $\mathcal{N}$ be the set of nodes, $\mathcal{P}$ predicates, and $\mathcal{G}$ graphs. A locational index of depth i is $I_L^i(n) = \langle \gamma, \eta \rangle$ s.t. $\gamma : \mathcal{P} \rightarrow 2^{\mathcal{G}}$ and $\eta : \mathcal{G} \rightarrow 2^{\mathcal{N}}$

Given a *graph/fragment*, identify which *nodes* contain it



Graphs	Nodes
	$\{n_1, n_5, n_6\}$
	$\{n_1, n_2, n_7\}$
	$\{n_1, n_6\}$
	$\{n_2, n_4, n_5\}$
	$\{n_4, n_7\}$

LOCATIONAL INDEXES

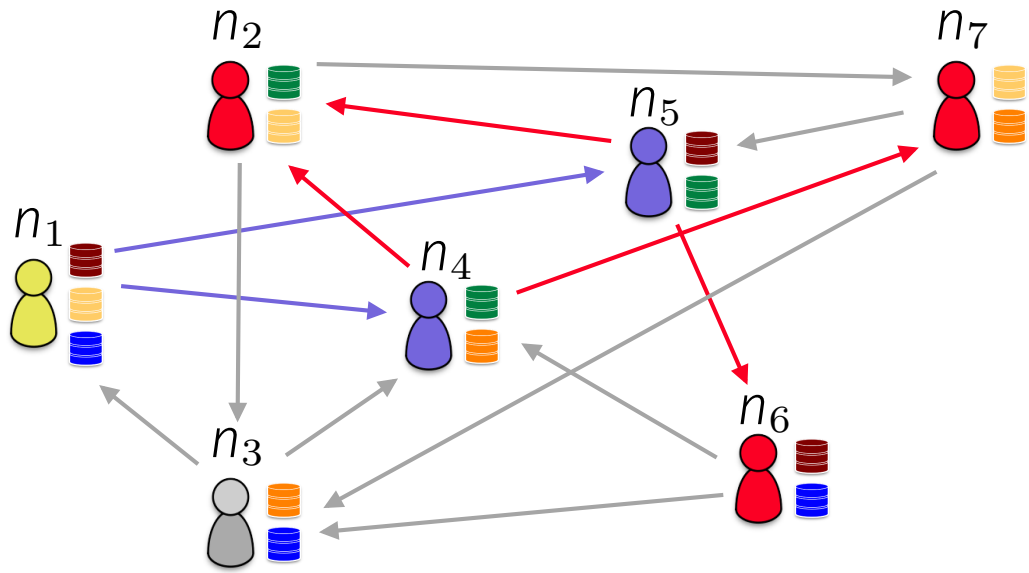
Identify *relevant* nodes to ask

Locational Index

Let $\mathcal{N}$ be the set of nodes, $\mathcal{P}$ predicates, and $\mathcal{G}$ graphs. A locational index of depth i is $I^i_L(n) = \langle \gamma, \eta \rangle$ s.t. $\gamma : \mathcal{P} \to 2^{\mathcal{G}}$ and $\eta : \mathcal{G} \to 2^{\mathcal{N}}$

Nodes pre-compute a *local* index over a neighborhood by *flooding*

e.g. for *depth* $= 2$:



BLOOM FILTERS

We can do better

Some combination of fragments might not produce any *join results*


```
SELECT * WHERE {
?pr dbo:nationality dbr:United_States .
?pr dbo:party ?pa
}
```

⇓

and produce no join result

Solution

Index *subjects* and *objects* and determine if they overlap

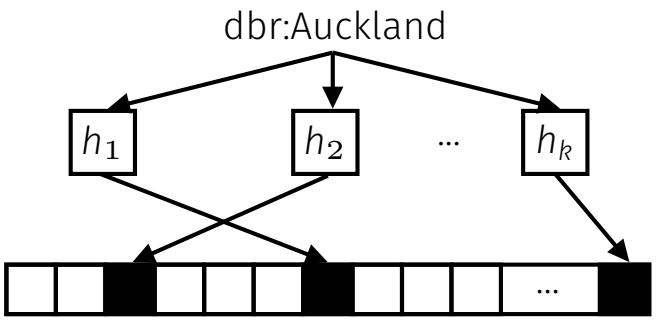
BLOOM FILTERS

Datasets are *too large* ⇒ Use *Bloom Filters*

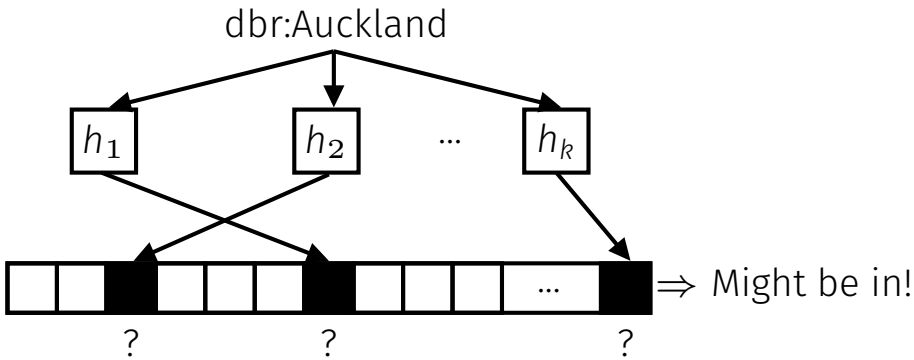
A Bloom Filter $\mathcal{B}$ consists of:

- A set of k hash functions $h_1, h_2, \ldots, h_k$
- A *bit vector* containing n bits

Insertion



Lookup



B.H. Bloom **Space/Time Trade-Offs in Hash Coding Algorithms with Allowable Errors** Commun. ACM 1970

BLOOM FILTERS

Problem

More *subjects/objects* $\Rightarrow$ more *bits* for a lower *false-positive* rate

$\sim 500M$ subjects/objects $\Rightarrow$ $4B$ bits for a false-positive rate of $< 0.1\%$

All bit vectors must be the same *size*

BLOOM FILTERS

Problem

More *subjects/objects* ⇒ more *bits* for a lower *false-positive* rate

$\sim 500M$ subjects/objects ⇒ $4B$ bits for a false-positive rate of $< 0.1\%$

All bit vectors must be the same *size*

Solution

One *bit vector* per URI prefix ⇒ *partitioned* filter

e.g. for **`http://dbpedia.org/resources/`**`Auckland`

BLOOM FILTERS

Problem

More *subjects/objects* ⇒ more *bits* for a lower *false-positive* rate

$\sim 500M$ subjects/objects ⇒ $4B$ bits for a false-positive rate of $< 0.1\%$

All bit vectors must be the same *size*

Solution

One *bit vector* per URI prefix ⇒ *partitioned* filter

e.g. for **`http://dbpedia.org/resources/`**`Auckland`

- Partitions require bit vectors of smaller sizes

  e.g. most partitions in our experiments $< 0.01\%$ false positives with $< 1M$ bits

BLOOM FILTERS

Problem

More *subjects/objects* ⇒ more *bits* for a lower *false-positive* rate

$\sim 500M$ subjects/objects ⇒ $4B$ bits for a false-positive rate of $< 0.1\%$

All bit vectors must be the same *size*

Solution

One *bit vector* per URI prefix ⇒ *partitioned* filter

e.g. for **`http://dbpedia.org/resources/`**`Auckland`

- Partitions require bit vectors of smaller sizes

  e.g. most partitions in our experiments $< 0.01\%$ false positives with $< 1M$ bits

- False positives are more *tolerable*

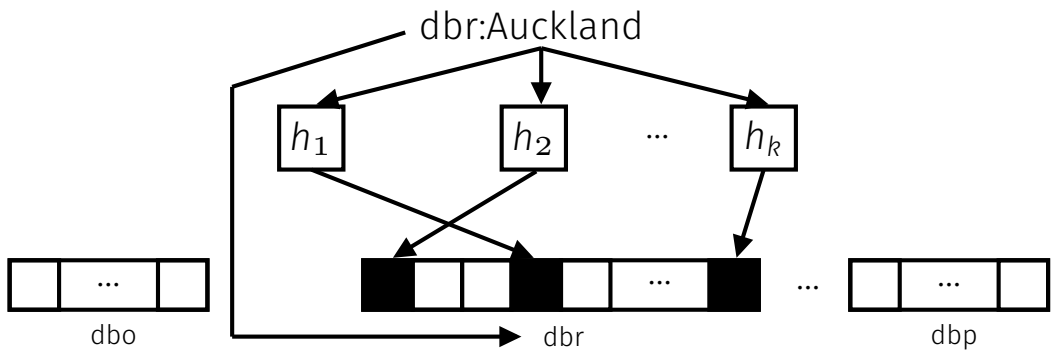
  URIs with *common prefixes* are likely to be in the same graph

PREFIX-PARTITIONED BLOOM FILTERS

Prefix-Partitioned Bloom Filter (PPBF)

A PPBF $\mathcal{B}^P$ is a 4-tuple $\mathcal{B}^P = \langle P, \hat{B}, \theta, H \rangle$ with:

- A set of prefixes P,
- A set of bit vectors $\hat{B}$,
- A prefix-mapping function $\theta : P \rightarrow \hat{B}$, and
- A set of hash functions H



$u \Subset \mathcal{B}^P$ means that u *may* be in $\mathcal{B}^P$

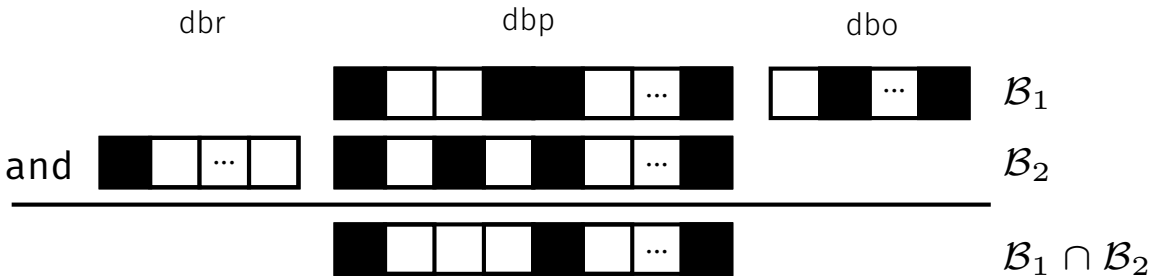
PREFIX-PARTITIONED BLOOM FILTERS

If the *intersection* of two PPBFs is empty, the fragments produce no *join results*

PPBF Intersection

The intersection of two PPBFs with the **same set** of hash functions and bit vectors of the **same size** is $\mathcal{B}_1^P \cap \mathcal{B}_2^P = \langle P_\cap, \hat{B}_\cap, \theta_\cap, H \rangle$ with:

- $P_\cap = \mathcal{B}_1^P.P \cap \mathcal{B}_2^P.P$,
- $\hat{B}_\cap = \{\mathcal{B}_1^P.\theta(p) \text{ and } \mathcal{B}_2^P.\theta(p) \mid p \in P_\cap\}$, and
- $\theta_\cap : P_\cap \to \hat{B}_\cap$



PREFIX-PARTITIONED BLOOM FILTER

- *Computation*: Download PPBFs for all reachable fragments

  And create PPBFs for local fragments

PREFIX-PARTITIONED BLOOM FILTER

- *Computation*: Download PPBFs for all reachable fragments

  And create PPBFs for local fragments

- *Usage*: Check intersection of PPBFs for joining triple patterns

  If a fragment overlaps with no other fragment, it is pruned

PREFIX-PARTITIONED BLOOM FILTER

- *Computation*: Download PPBFs for all reachable fragments

  And create PPBFs for local fragments
- *Usage*: Check intersection of PPBFs for joining triple patterns

  If a fragment overlaps with no other fragment, it is pruned
- *Accessing data*: Ask nodes with matched fragments directly

MATCHING PPBFS TO QUERIES

```
  SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
  }
```

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

MATCHING PPBFS TO QUERIES

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

 , have predicate `rdf:type`

`dbo:President` $\in$ 

`dbo:President` $\notin$ 

MATCHING PPBFS TO QUERIES

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

 has predicate `dbo:nationality`

`dbr:United_States` ∈ 

MATCHING PPBFS TO QUERIES

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

 have predicate **dbo : party**

MATCHING PPBFS TO QUERIES

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

tp2 and tp3 join on **?pr**

$\mathcal{B}^P(\text{🟠}) \cap \mathcal{B}^P(\text{🔵}) = \emptyset$

🟠 and 🔵 produce no join result

MATCHING PPBFS TO QUERIES

```
  SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
  }
```

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

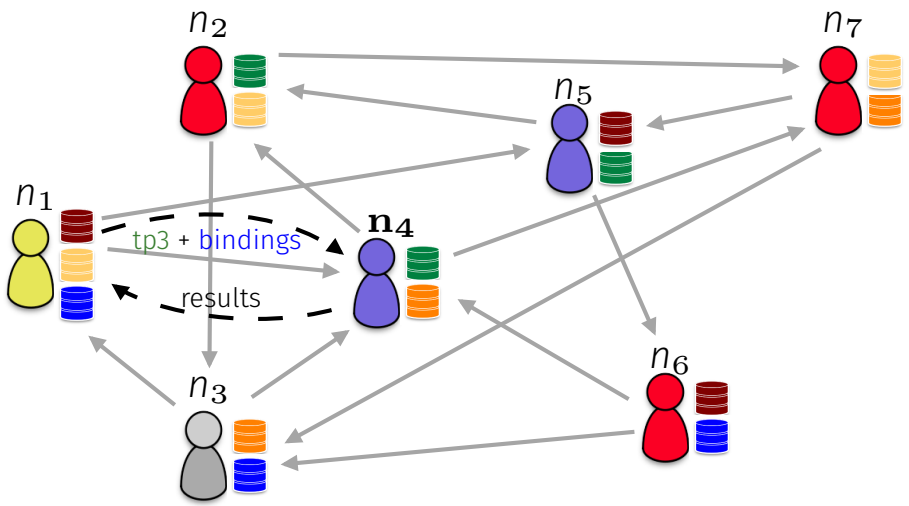
dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

We eliminated  and 

Fewer nodes have to be queried

QUERY PROCESSING - PPBF INDEX

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```



Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

Graphs	Nodes
	$\{n_1, n_5, n_6\}$
	$\{n_1, n_2, n_7\}$
	$\{n_1, n_6\}$
	$\{n_2, \mathbf{n_4}, n_5\}$
	$\{n_4, n_7\}$

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

Evaluate: tp3

Bindings for tp1 ⋈ tp2:

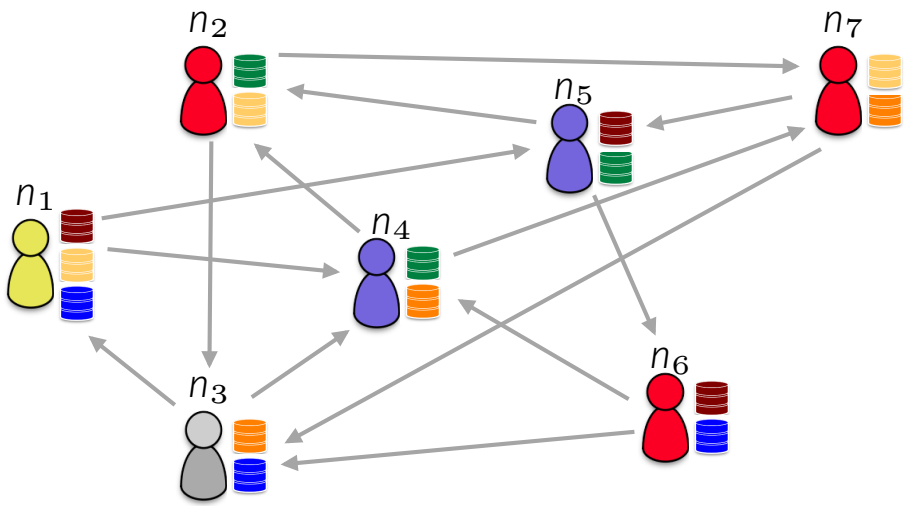
{{?pr=dbr:Donald_Trump}
{?pr=dbr:Barack_Obama}}

Result:

{{?pr=dbr:Donald_Trump,
?pa=dbr:Republican_Party},
{?pr=dbr:Barack_Obama,
?pa=dbr:Democratic_Party}}

QUERY PROCESSING - PPBF INDEX

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```



dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

Graphs	Nodes
	$\{n_1, n_5, n_6\}$
	$\{n_1, n_2, n_7\}$
	$\{n_1, n_6\}$
	$\{n_2, n_4, n_5\}$
	$\{n_4, n_7\}$

Result:

```
{{?pr=dbr:Donald_Trump,
?pa=dbr:Republican_Party},
{?pr=dbr:Barack_Obama,
?pa=dbr:Democratic_Party}}
```

EXPERIMENTAL SETUP

- Implemented on top of *PIQNIC*[2]

[2] C. Aebeloe et al. **A Decentralized Architecture for Sharing and Querying Semantic Data.** ESWC 2019

[3] A. Hasnain et al. **Extending LargeRDFBench for Multi-Source Data at Scale for SPARQL Endpoint Federation.** SSWS 2018

EXPERIMENTAL SETUP

- Implemented on top of *PIQNIC*[2]
- 4xAMD Opteron 6376 16 cores, 2.3GHz each, 516GB RAM
  - *200 nodes* running concurrently

[2] C. Aebeloe et al. **A Decentralized Architecture for Sharing and Querying Semantic Data.** ESWC 2019

[3] A. Hasnain et al. **Extending LargeRDFBench for Multi-Source Data at Scale for SPARQL Endpoint Federation.** SSWS 2018

EXPERIMENTAL SETUP

- Implemented on top of *PIQNIC*[2]
- 4xAMD Opteron 6376 16 cores, 2.3GHz each, 516GB RAM
  - *200 nodes* running concurrently
- LargeRDFBench[3]: *Benchmark* suite for federated query engines

[2] C. Aebeloe et al. **A Decentralized Architecture for Sharing and Querying Semantic Data.** ESWC 2019

[3] A. Hasnain et al. **Extending LargeRDFBench for Multi-Source Data at Scale for SPARQL Endpoint Federation.** SSWS 2018

EXPERIMENTAL SETUP

- Implemented on top of *PIQNIC*[2]
- 4xAMD Opteron 6376 16 cores, 2.3GHz each, 516GB RAM
  - *200 nodes* running concurrently
- LargeRDFBench[3]: *Benchmark* suite for federated query engines
- 13 datasets with a total of over *1B triples*

[2] C. Aebeloe et al. **A Decentralized Architecture for Sharing and Querying Semantic Data.** ESWC 2019

[3] A. Hasnain et al. **Extending LargeRDFBench for Multi-Source Data at Scale for SPARQL Endpoint Federation.** SSWS 2018

EXPERIMENTAL SETUP

- Implemented on top of *PIQNIC*[2]
- 4xAMD Opteron 6376 16 cores, 2.3GHz each, 516GB RAM
  - *200 nodes* running concurrently
- LargeRDFBench[3]: *Benchmark* suite for federated query engines
- 13 datasets with a total of over *1B triples*
- *40* queries in 4 different categories
  - Simple (*S*): 14 queries
  - Complex (*C*): 10 queries
  - Larga-Data (*L*): 8 queries
  - Complex and High Amounts of Data Sources (*CH*): 8 queries

[2] C. Aebeloe et al. **A Decentralized Architecture for Sharing and Querying Semantic Data.** ESWC 2019

[3] A. Hasnain et al. **Extending LargeRDFBench for Multi-Source Data at Scale for SPARQL Endpoint Federation.** SSWS 2018

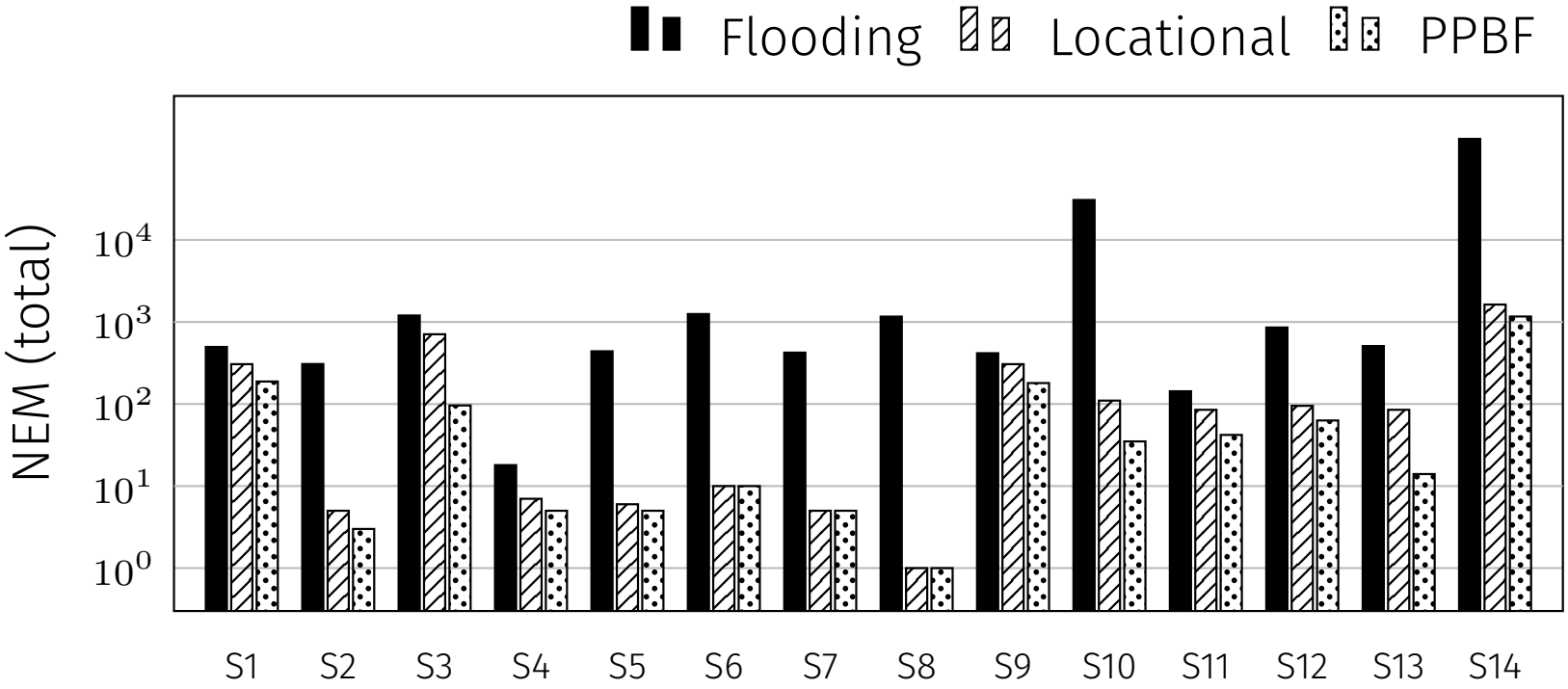
EXPERIMENTAL SETUP

- Implemented on top of *PIQNIC*[2]
- 4xAMD Opteron 6376 16 cores, 2.3GHz each, 516GB RAM
  - *200 nodes* running concurrently
- LargeRDFBench[3]: *Benchmark* suite for federated query engines
- 13 datasets with a total of over *1B triples*
- *40* queries in 4 different categories
  - Simple (*S*): 14 queries
  - Complex (*C*): 10 queries
  - Larga-Data (*L*): 8 queries
  - Complex and High Amounts of Data Sources (*CH*): 8 queries
- $ttl = 5$, $replication = 5\%$, $neighbors = 5$

[2] C. Aebeloe et al. **A Decentralized Architecture for Sharing and Querying Semantic Data.** ESWC 2019

[3] A. Hasnain et al. **Extending LargeRDFBench for Multi-Source Data at Scale for SPARQL Endpoint Federation.** SSWS 2018

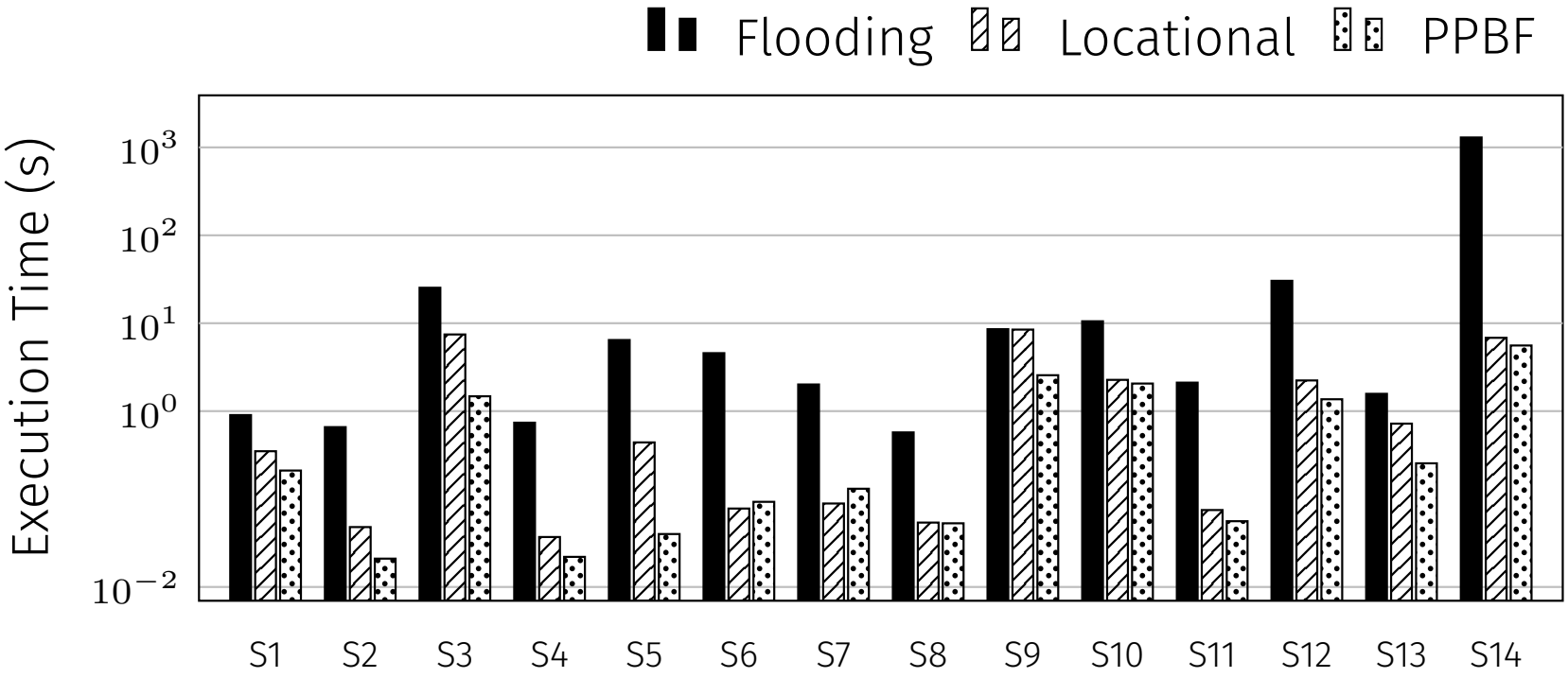
NUMBER OF EXCHANGED MESSAGES



Both indexes result in *fewer messages*, since nodes are asked *directly*

PPBF indexes *further reduce* the number of messages by *eliminating* irrelevant fragments/nodes

QUERY EXECUTION TIME



Both *Locational* and *PPBF* indexes improve performance, PPBF indexes *more significantly*

SUMMARY

- *Decentralized indexes*: Locational indexes & PPBFs

SUMMARY

- *Decentralized indexes*: Locational indexes & PPBFs
- *Implemented* on top of a decentralized system: PIQNIC

SUMMARY

- *Decentralized indexes*: Locational indexes & PPBFs
- *Implemented* on top of a decentralized system: PIQNIC
- *Performance gains*: up to
  - Locational index: 186x, PPBFs: 233x

SUMMARY

- *Decentralized indexes*: Locational indexes & PPBFs
- *Implemented* on top of a decentralized system: PIQNIC
- *Performance gains*: up to
  - Locational index: 186x, PPBFs: 233x
- *Traffic reduction*: up to
  - Locational index: 279x, PPBFs: 876x

SUMMARY

- *Decentralized indexes*: Locational indexes & PPBFs
- *Implemented* on top of a decentralized system: PIQNIC
- *Performance gains*: up to
  - Locational index: 186x, PPBFs: 233x
- *Traffic reduction*: up to
  - Locational index: 279x, PPBFs: 876x
- PPBFs enabled *answering* queries that *timed out* with alternative approaches

SUMMARY

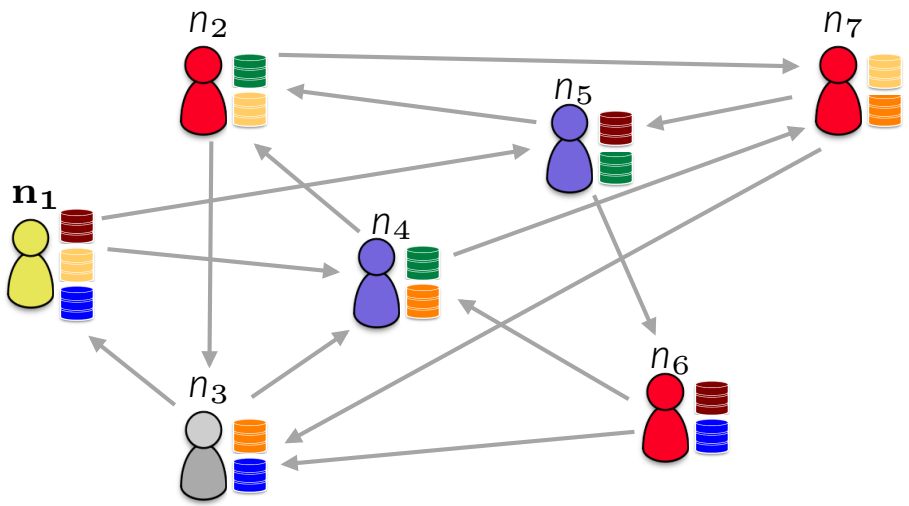
- *Decentralized indexes*: Locational indexes & PPBFs
- *Implemented* on top of a decentralized system: PIQNIC
- *Performance gains*: up to
  - Locational index: 186x, PPBFs: 233x
- *Traffic reduction*: up to
  - Locational index: 279x, PPBFs: 876x
- PPBFs enabled *answering* queries that *timed out* with alternative approaches

For more information:
`http://relweb.cs.aau.dk/ppbfs`

Thank you!

QUERY PROCESSING - LOCATIONAL INDEX

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```



dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

Graphs	Nodes
	$\{\mathbf{n_1}, n_5, n_6\}$
	$\{\mathbf{n_1}, n_2, n_7\}$
	$\{n_1, n_6\}$
	$\{n_2, n_4, n_5\}$
	$\{n_4, n_7\}$

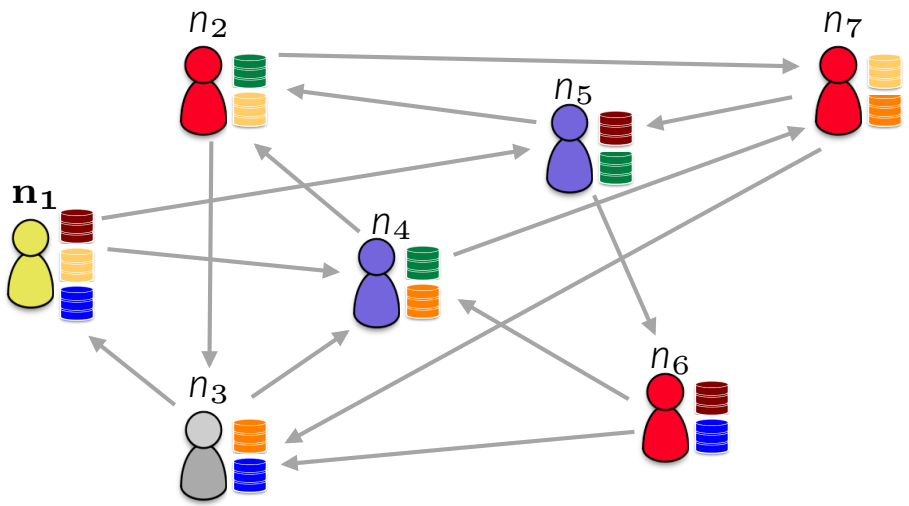
Evaluate: tp1

Result:

```
{{?pr=dbr:Donald_Trump},
{?pr=dbr:Barack_Obama}}
```

QUERY PROCESSING - LOCATIONAL INDEX

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```



dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

Graphs	Nodes
	$\{n_1, n_5, n_6\}$
	$\{n_1, n_2, n_7\}$
	$\{\mathbf{n_1}, n_6\}$
	$\{n_2, n_4, n_5\}$
	$\{n_4, n_7\}$

Evaluate: tp2

Bindings:

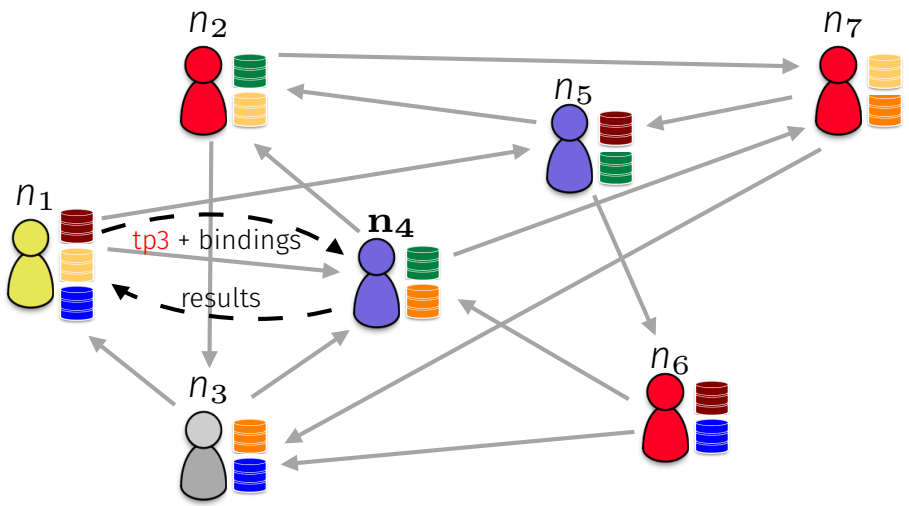
{{?pr=dbr:Donald_Trump},
{?pr=dbr:Barack_Obama}}

Result:

{{?pr=dbr:Donald_Trump},
{?pr=dbr:Barack_Obama}}

QUERY PROCESSING - LOCATIONAL INDEX

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```



Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

Graphs	Nodes
	$\{n_1, n_5, n_6\}$
	$\{n_1, n_2, n_7\}$
	$\{n_1, n_6\}$
	$\{n_2, \mathbf{n_4}, n_5\}$
	$\{\mathbf{n_4}, n_7\}$

dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

Evaluate: tp3

Bindings:

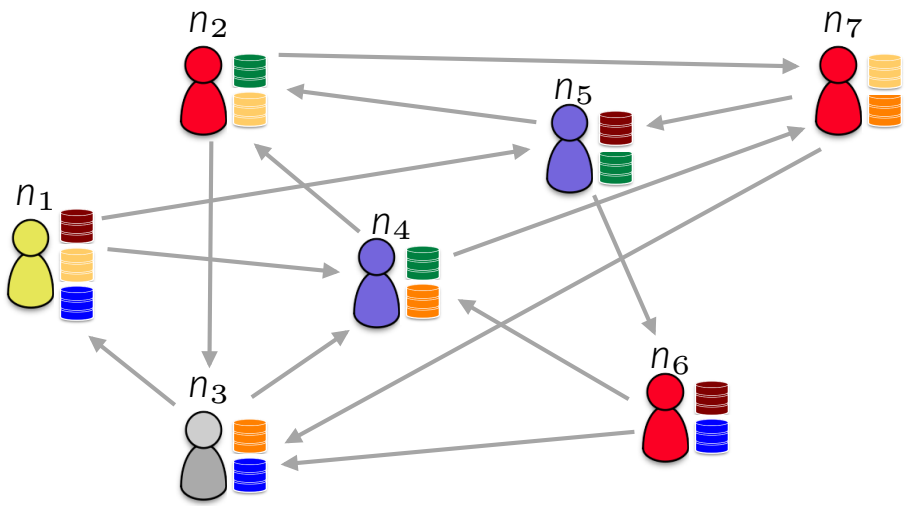
```
{{?pr=dbr:Donald_Trump},
{?pr=dbr:Barack_Obama}}
```

Result:

```
{{?pr=dbr:Donald_Trump,
?pa=dbr:Republican_Party},
{?pr=dbr:Barack_Obama,
?pa=dbr:Democratic_Party}}
```

QUERY PROCESSING - LOCATIONAL INDEX

```
SELECT * WHERE {
tp1 ?pr rdf:type dbo:President .
tp2 ?pr dbo:nationality dbr:United_States .
tp3 ?pr dbo:party ?pa
}
```



dbr:Donald_Trump rdf:type dbo:President
dbr:Barack_Obama rdf:type dbo:President

dbr:Mette_Frederiksen rdf:type dbo:Prime_Minister
dbr:Jacinda_Ardern rdf:type dbo:Prime_Minister

dbr:Donald_Trump dbo:nationality dbr:United_States
dbr:Barack_Obama dbo:nationality dbr:United_States
dbr:Angela_Merkel dbo:nationality dbr:Germany
dbr:Emmanuel_Macron dbo:nationality dbr:France

dbr:Donald_Trump dbo:party dbr:Republican_Party
dbr:Barack_Obama dbo:party dbr:Democratic_Party

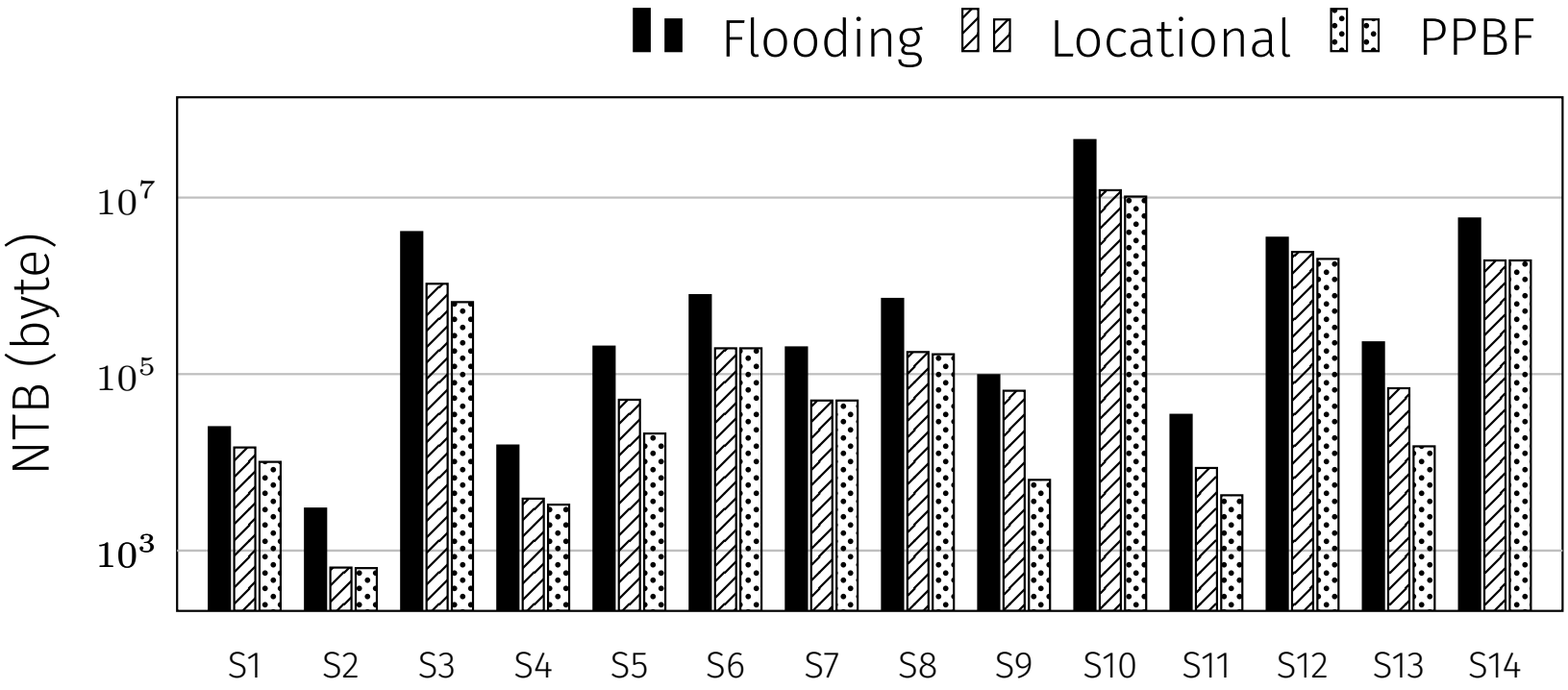
dbr:Mette_Frederiksen dbo:party dbr:Socialdemokratiet
dbr:Jacinda_Ardern dbo:party dbr:Labour

Predicates	Graphs
rdf:type	
dbo:nationality	
dbo:party	

Graphs	Nodes
	$\{n_1, n_5, n_6\}$
	$\{n_1, n_2, n_7\}$
	$\{n_1, n_6\}$
	$\{n_1, n_2, n_4\}$
	$\{n_4, n_7\}$

```
{{?pr=dbr:Donald_Trump,
?pa=dbr:Republican_Party},
{?pr=dbr:Barack_Obama,
?pa=dbr:Democratic_Party}}
```

NUMBER OF TRANSFERRED BYTES (NTB)



Since *fewer* fragments are queried, the amount of transferred data is *reduced*