جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

IBM

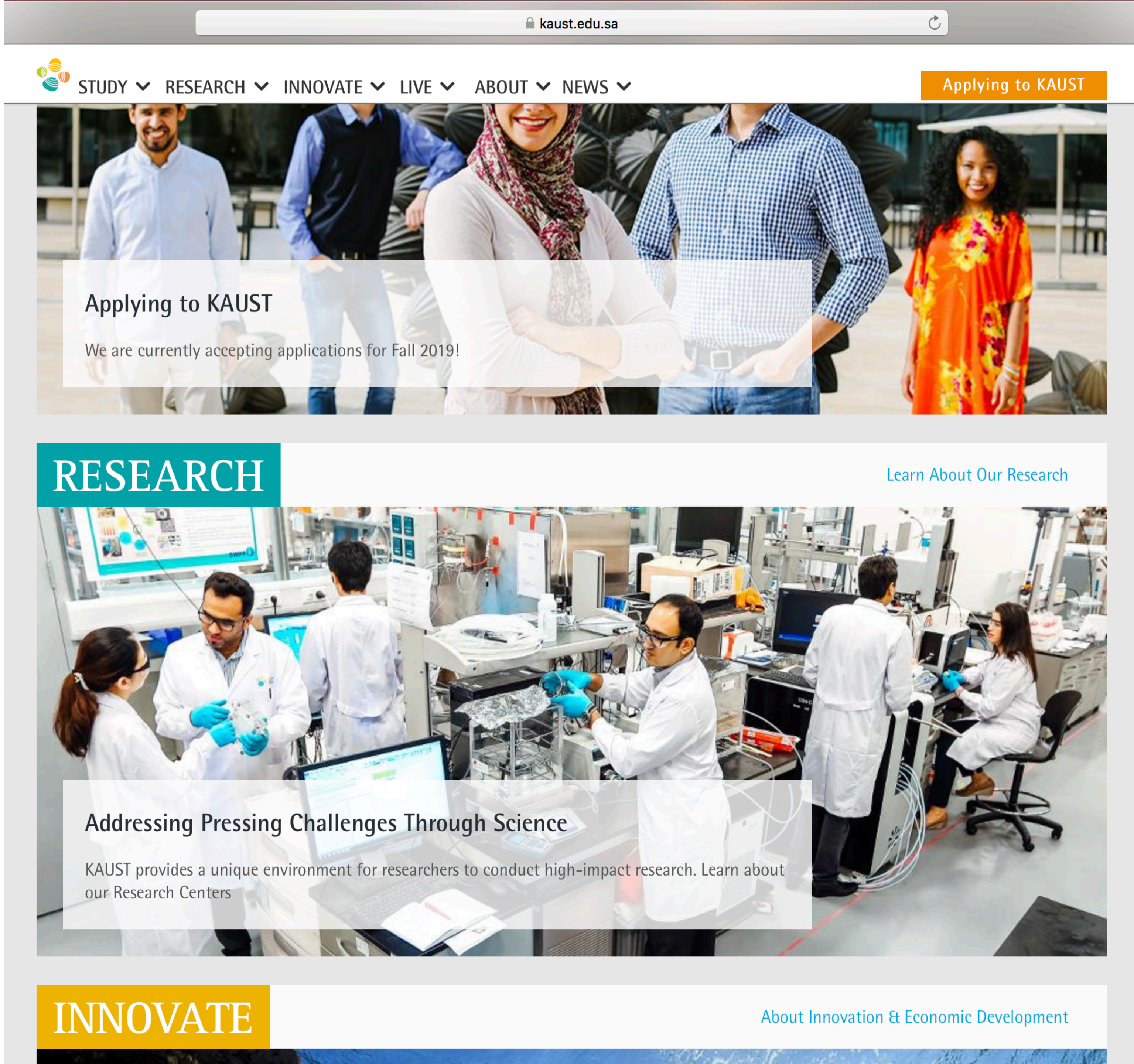
How to build scalable **SPARQL** engines for Big **RDF** data

Panos Kalnis

King Abdullah University of Science and Technology (KAUST)

KAUST.edu.SA


3 hours
Saudi Arabia


kaust.edu.sa
STUDY
RESEARCH
INNOVATE
LIVE
ABOUT
NEWS
Applying to KAUST
Applying to KAUST
We are currently accepting applications for Fall 2019!
RESEARCH
Learn About Our Research
Addressing Pressing Challenges Through Science
KAUST provides a unique environment for researchers to conduct high-impact research. Learn about our Research Centers
INNOVATE
About Innovation & Economic Development

Info: **Panos.Kalnis**@kaust.edu.sa

- Paid internships for undergrads
- Scholarships for Master’s & PhDs
- PostDoc
- Faculty positions

- Big Data, Analytics
- Cloud, Parallel, Distributed, HPC
- Machine Learning, **AI**
- Visual Computing
- Bio-Informatics
- Cyber-Security

Systems for Big Data & Machine Learning

Greece Italy USA France S. Arabia Egypt Taiwan Belgium Mexico India China

Systems for Big Data & Machine Learning

Materials Science
Hang

Software Engineering
Chen-Yu Kostas

Distributed Systems
Marco

Big Data
Panos

Algorithms
Atal

Mathematical Optimization
Aritra Elhoucine

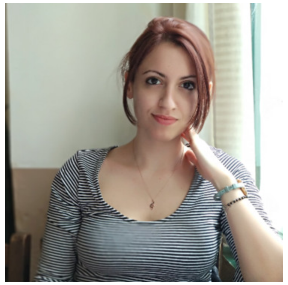
Networks
Ahmed

Paid **Remote** Internships

- Final year undergraduates, or Master's
- 3 – 6 months, start anytime
- US$ **1000 / month**
- Meetings via Zoom
- Can be combined with final year project
  - **Διπλωματική / Πτυχιακή εργασία**

Info: **Panos.Kalnis**@kaust.edu.sa

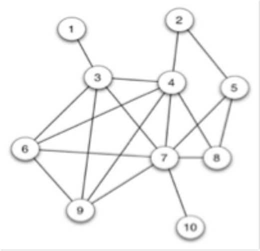
Klearchos Kosmanos
MSc – Aristotle Univ. Thessaloniki



Kelly Kostopoulou
PhD – Columbia Univ. New York

Stamatis Anoustis
PhD - KAUST

Graphs are Everywhere…



PubChem

UniProt

…But processing is expensive

- E.g., subgraph isomorphism, NP

"Big": CPU vs. Big size

RDF Data

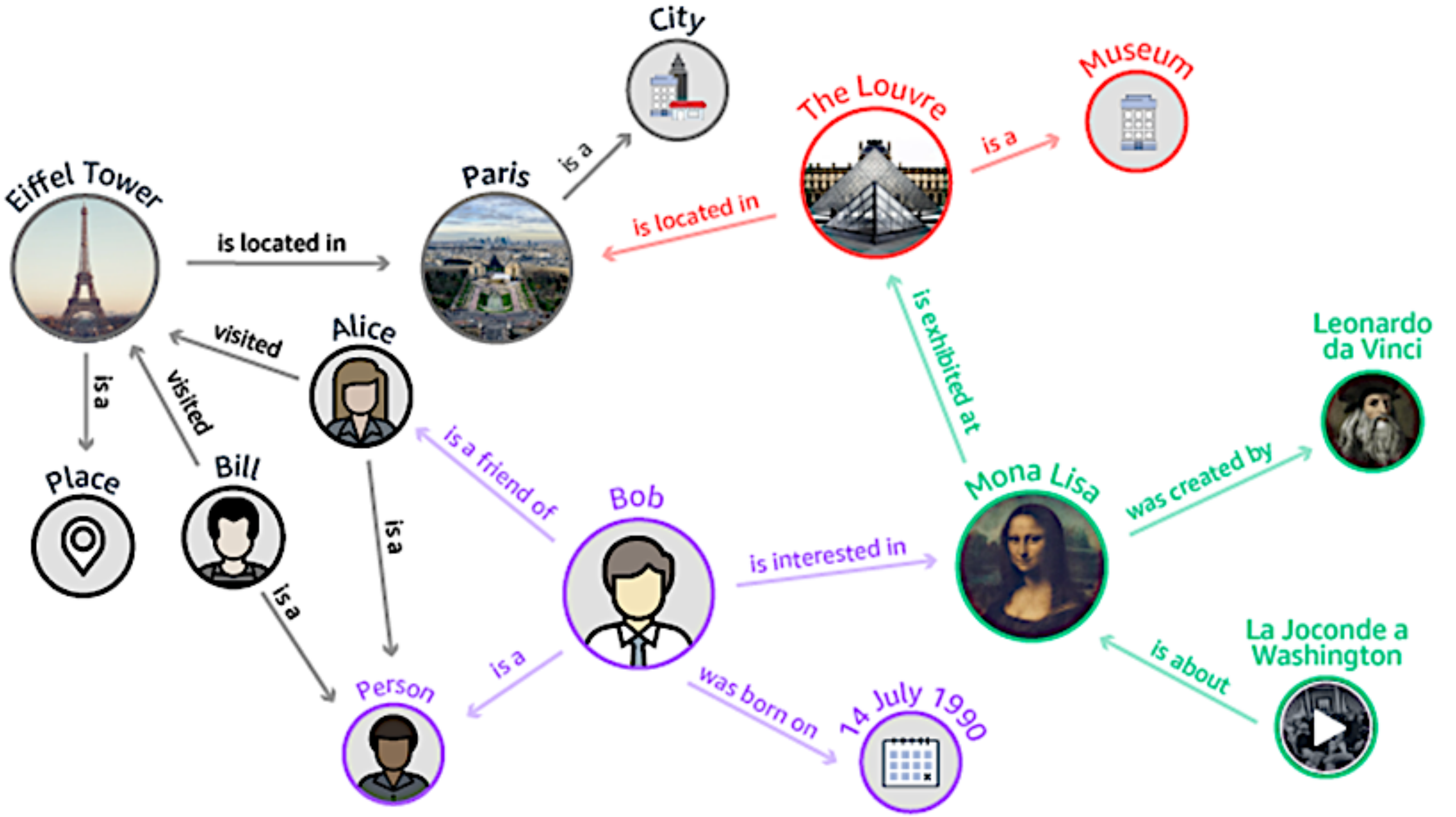


Image from: https://aws.amazon.com/blogs/apn/exploring-knowledge-graphs-on-amazon-neptune-using-metaphactory/

BIO2RDF

11 Billion

DBpedia

23 Billion

PubChem

130 Billion

RDF and SPARQL

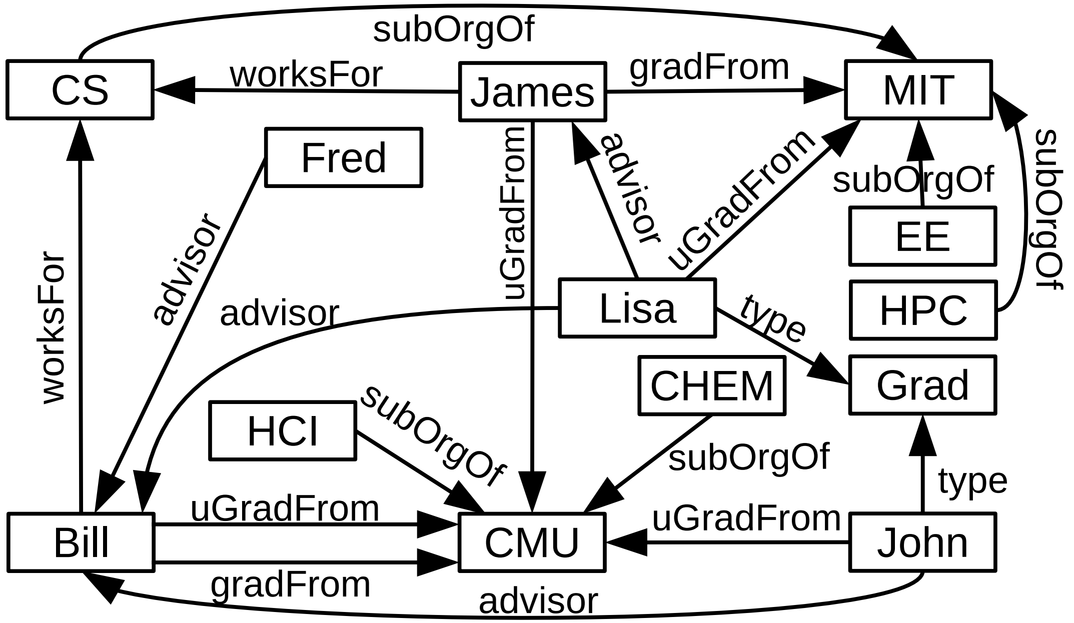
RDF: Set of triples:

< **S**ubject	**P**redicate	**O**bject >
James	gradFrom	MIT
EE	subOrgOf	MIT
James	worksFor	CS
CS	subOrgOf	MIT
Lisa	advisor	Bill
John	advisor	Bill
	…	
	…	

RDF and SPARQL

RDF: Set of triples: or Directed edge-labelled graph

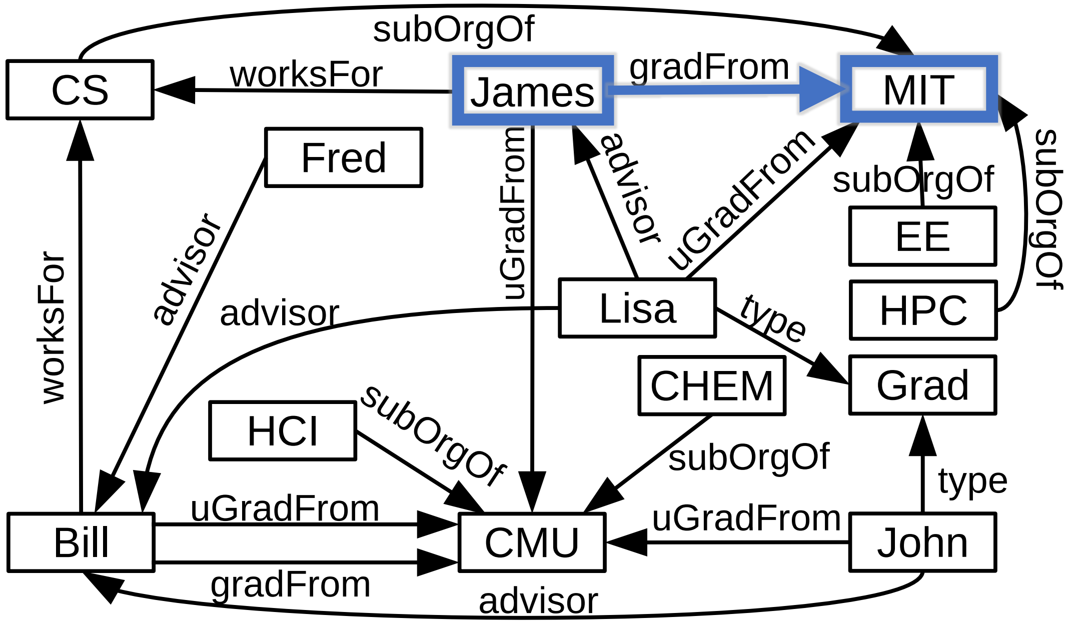
< **S**ubject	**P**redicate	**O**bject >
James	gradFrom	MIT
EE	subOrgOf	MIT
James	worksFor	CS
CS	subOrgOf	MIT
Lisa	advisor	Bill
John	advisor	Bill
	…	
	…	



RDF and SPARQL

RDF: Set of triples: or Directed edge-labelled graph

< **S**ubject	**P**redicate	**O**bject >
James	gradFrom	MIT
EE	subOrgOf	MIT
James	worksFor	CS
CS	subOrgOf	MIT
Lisa	advisor	Bill
John	advisor	Bill
	…	
	…	



RDF and SPARQL

SPARQL: query language for RDF

```
SELECT ?prof ?stud WHERE {
   ?prof   worksFor   CS     .
   ?stud   advisor    ?prof  .
}
```

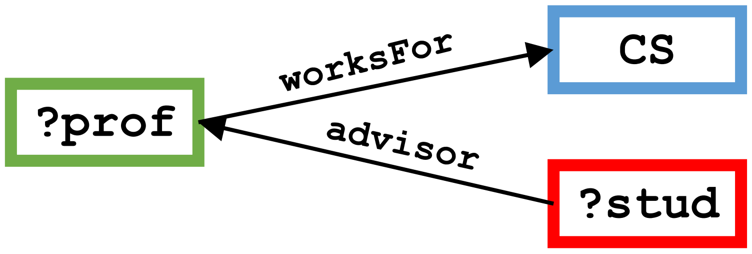
Return all professors who work for CS department, and their students

RDF and SPARQL

SPARQL: query language for RDF

```
SELECT ?prof ?stud WHERE {
   ?prof   worksFor   CS     .
   ?stud   advisor    ?prof  .
}
```

Return all professors who work for CS department, and their students

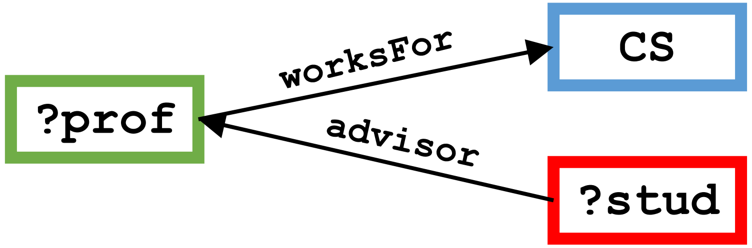


RDF and SPARQL

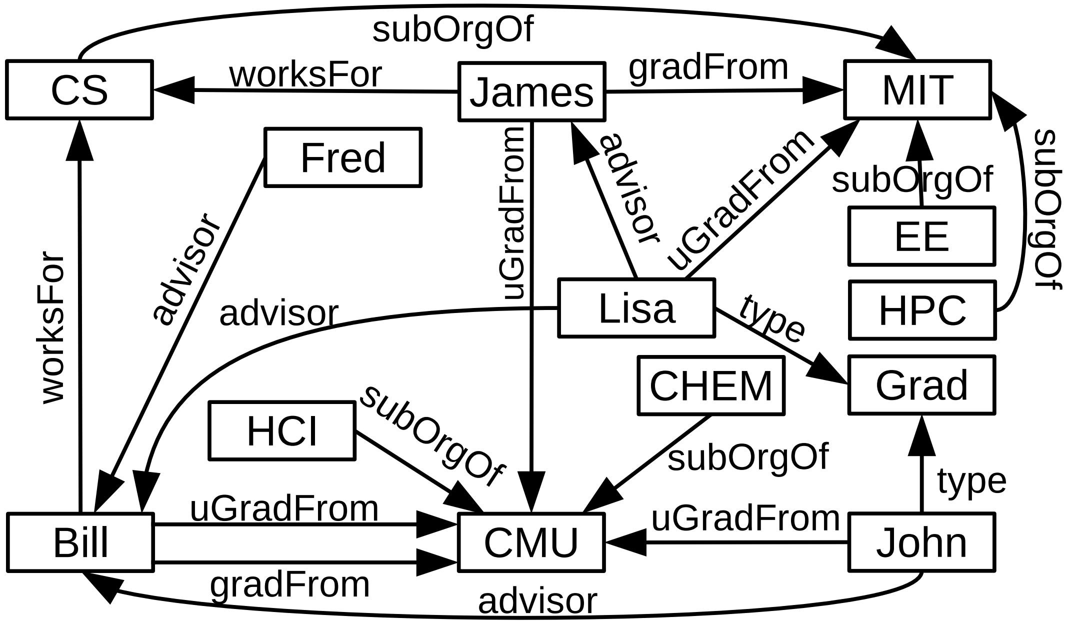
SPARQL: query language for RDF

```
SELECT ?prof ?stud WHERE {
   ?prof   worksFor   CS      .
   ?stud   advisor    ?prof   .
}
```

Return all professors who work for CS department, and their students



RDF: Directed edge-labelled graph

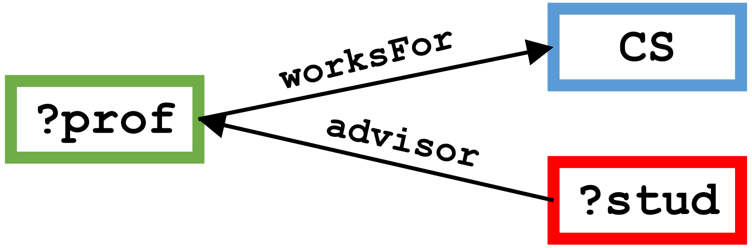


RDF and SPARQL

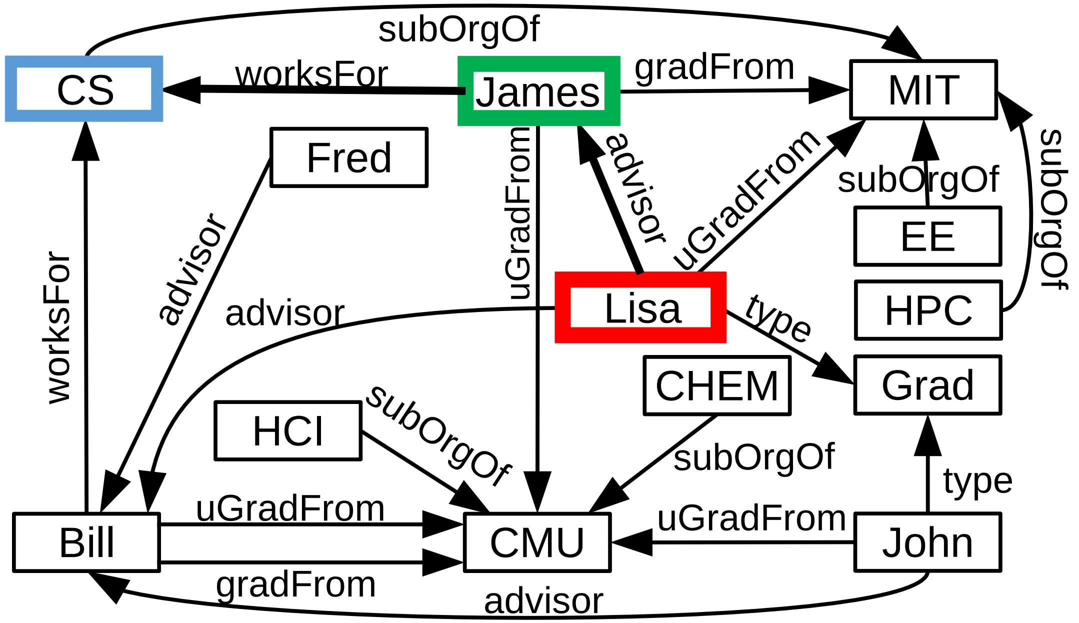
SPARQL: query language for RDF

```
SELECT ?prof ?stud WHERE {
   ?prof   worksFor   CS      .
   ?stud   advisor    ?prof   .
}
```

Return all professors who work for CS department, and their students



RDF: Directed edge-labelled graph

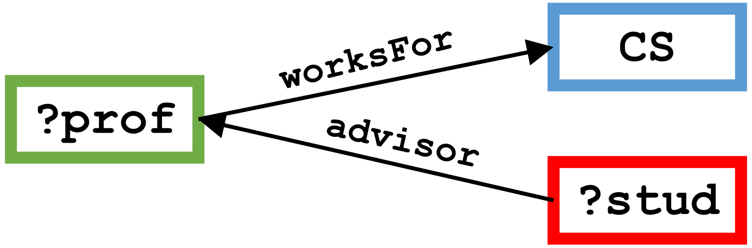


RDF and SPARQL

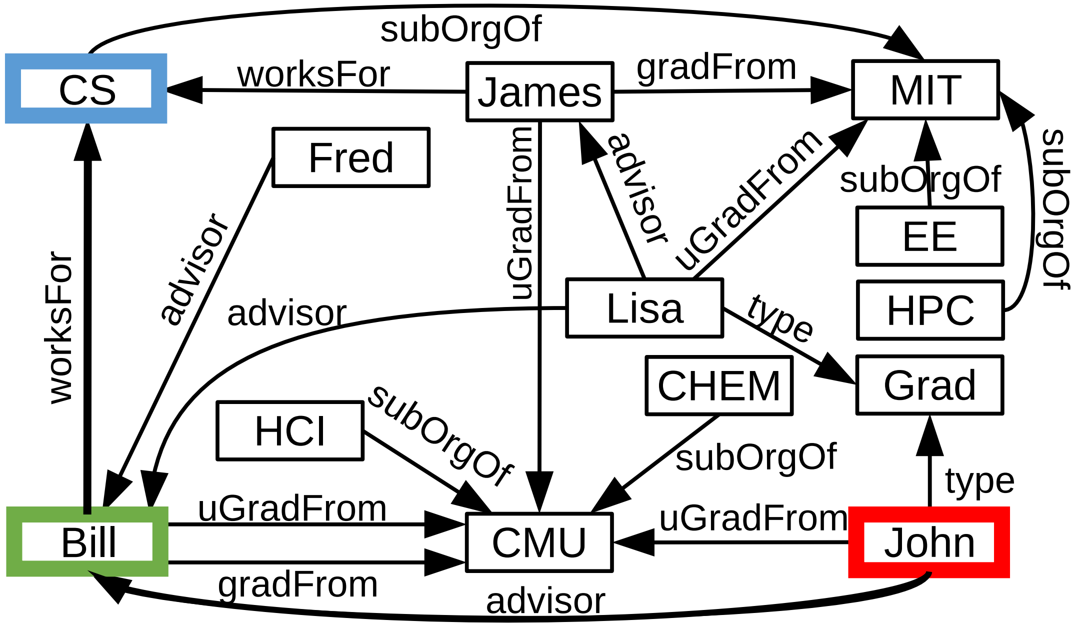
SPARQL: query language for RDF

```
SELECT ?prof ?stud WHERE {
   ?prof   worksFor   CS      .
   ?stud   advisor    ?prof   .
}
```

Return all professors who work for CS department, and their students



RDF: Directed edge-labelled graph

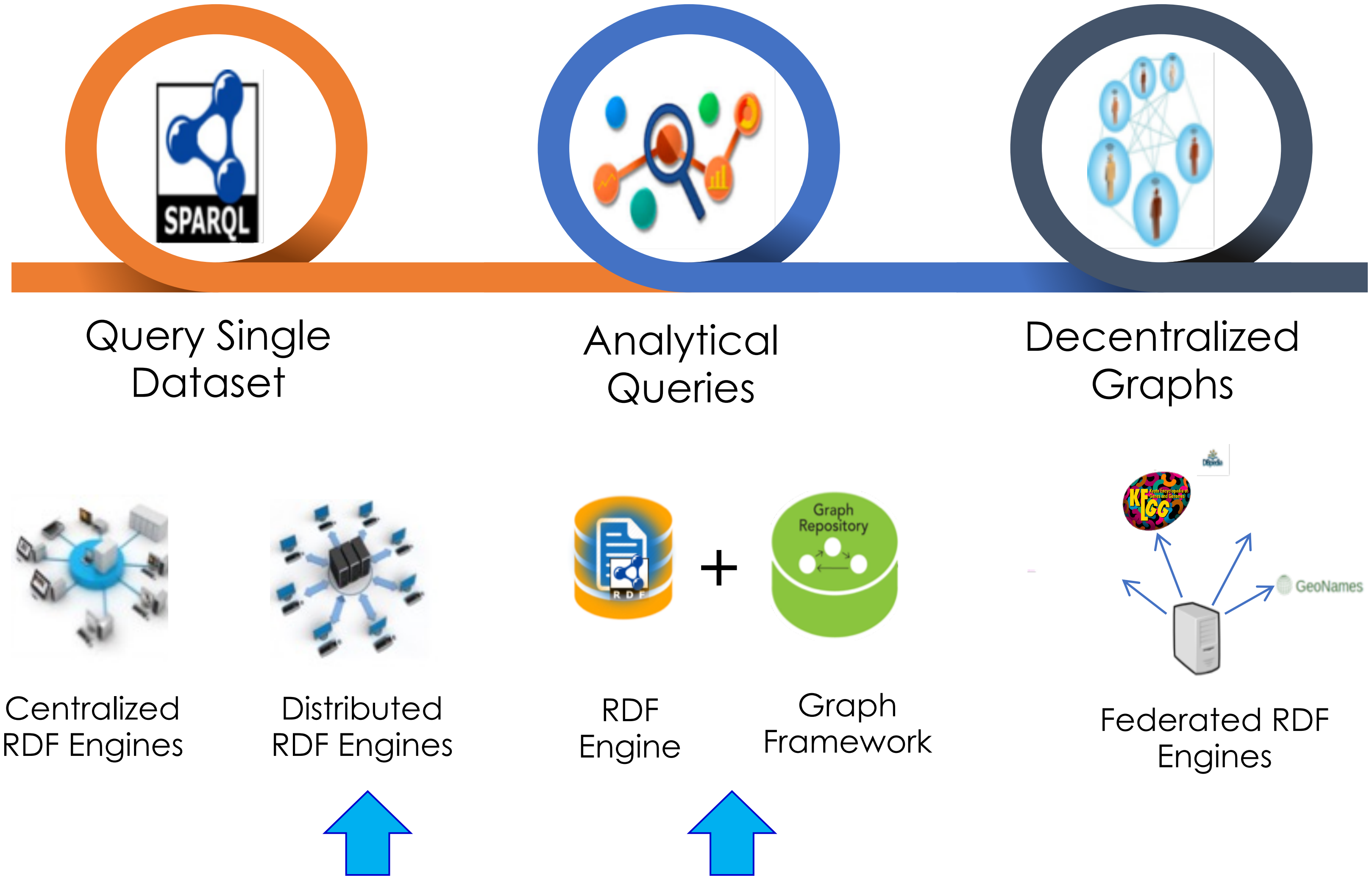


AdPart: Dynamic Partitioning

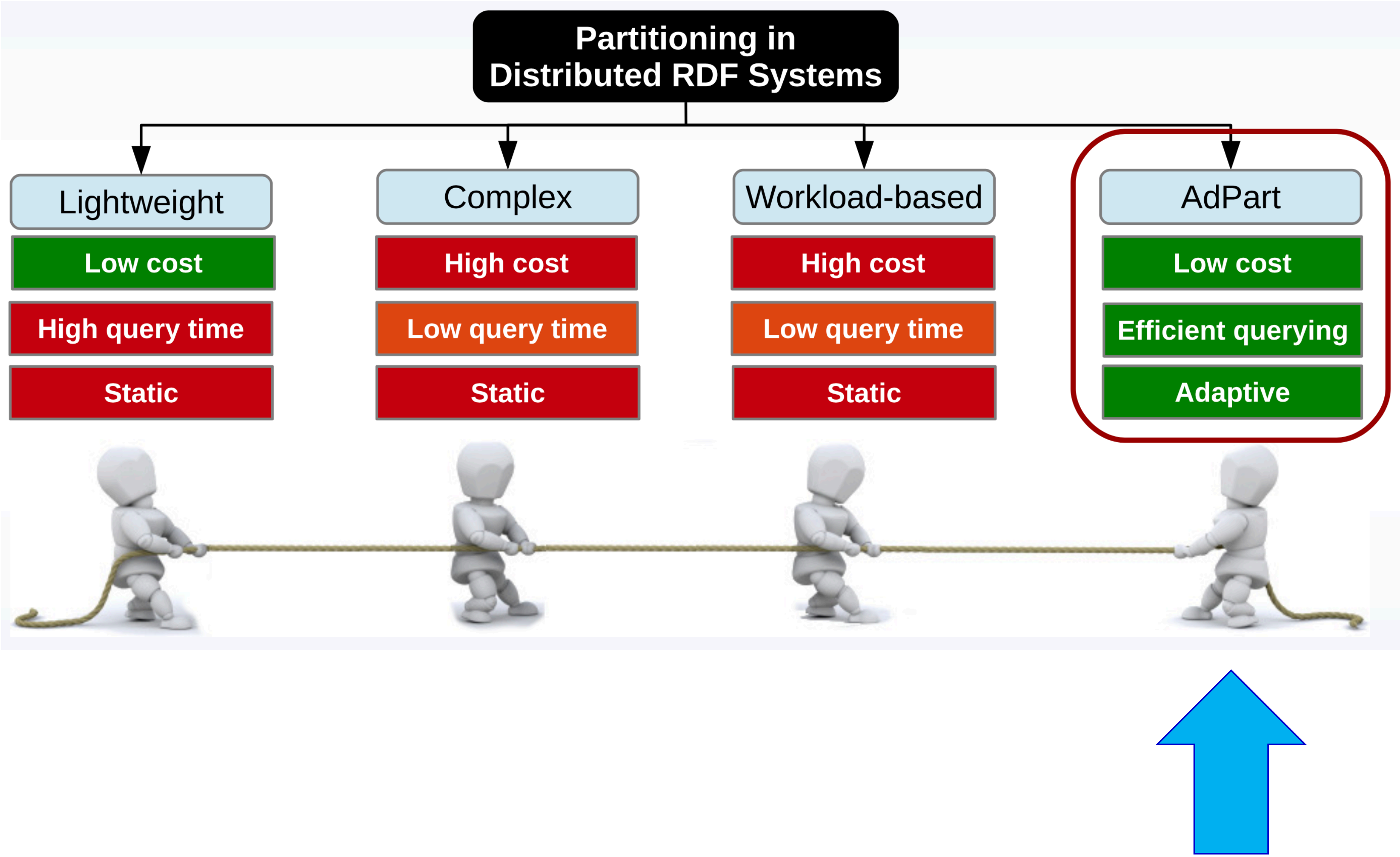
- *In: VLDB Journal, 2016*

- Evaluating SPARQL queries on massive RDF datasets, **PVLDB**, 2015
- Survey & experimental comparison of distributed SPARQL engines for very large RDF data, **PVLDB**, 2017
- Query optimizations over decentralized RDF graphs, **ICDE**, 2017
- Lusail: a system for querying linked data at scale, **PVDLB**, 2017
- A demonstration of Lusail: Querying linked data at scale, **(demo) SIGMOD**, 2017

Classification of RDF systems

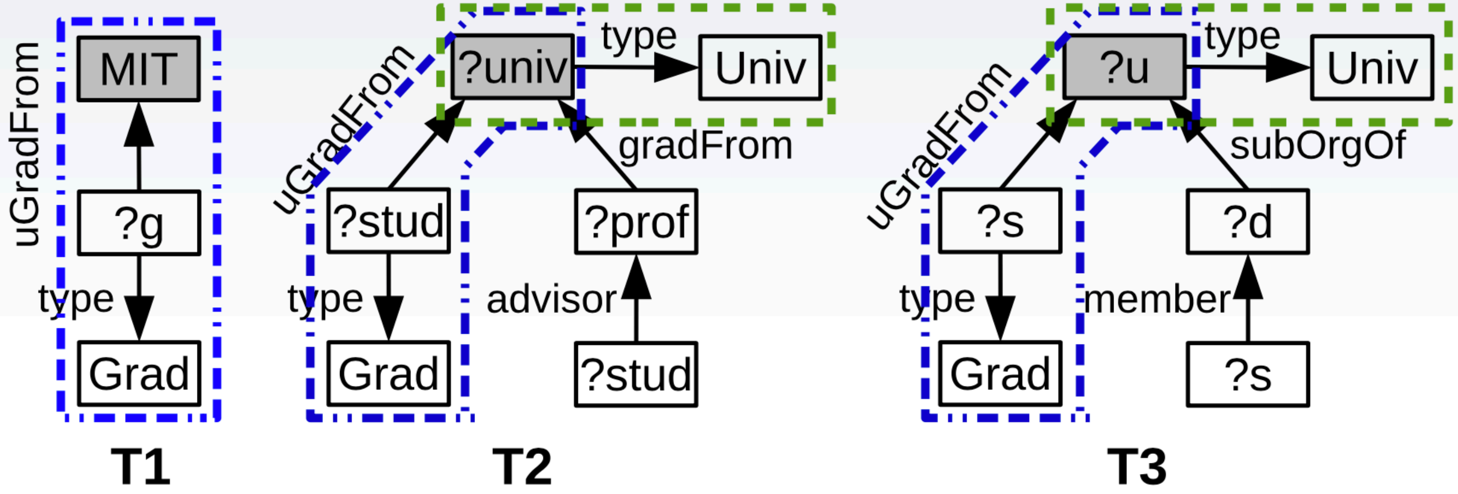


Partitioning for Parallel Processing

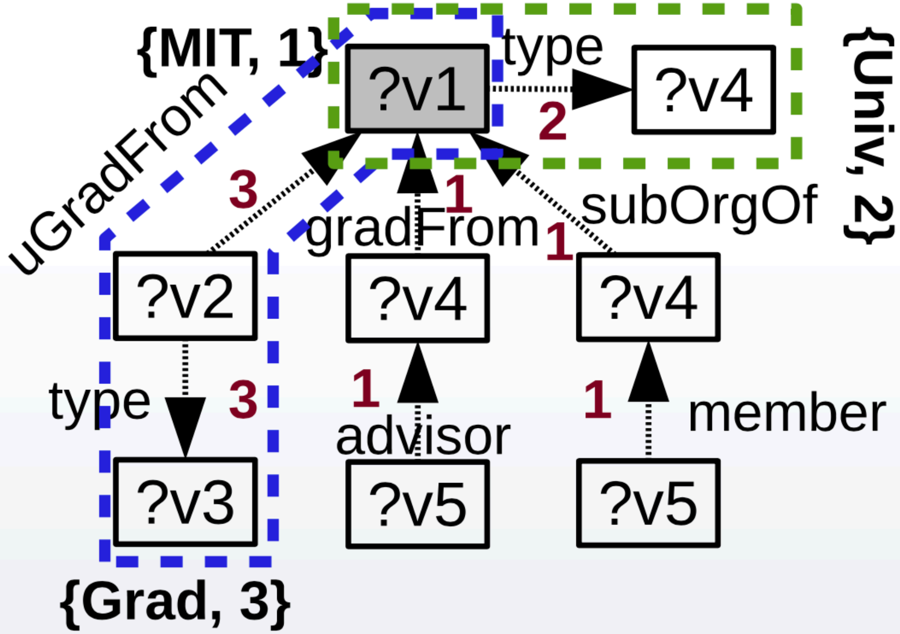


AdPart – Dynamic Partitioning

Queries decomposition



Updated heatmap



AdPart is at least 10x faster

Table 6: Runtime for LUBM-10240 queries (ms). SM: Single Machine, MR: MapReduce, and SS: Specialized systems. S2X failed to execute all queries; gStore and gStoreD could not preprocess the data within 24 hours.

	LUBM-10240	Complex Queries				Simple Queries			Geo-Mean	Query/h
		L1	L2	L3	L7	L4	L5	L6		
SM	RDF-3X	1,812,250	101,750	1,898,500	98,250	38	20	526	10,466	6
MR	SHARD	413,720	187,310	N/A	469,340	358,200	116,620	209,800	261,362	N/A
	H2RDF+	285,430	71,720	264,780	180,320	24,120	4,760	22,910	59,275	30
	CliqueSquare	125,020	71,010	80,010	224,040	90,010	24,000	37,010	74,488	39
	S2RDF-VP	217,537	28,917	145,761	29,965	46,770	5,233	11,308	35,845	52
	S2RDF-ExtVP	46,552	35,802	21,533	47,343	9,222	2,718	4,555	15,275	150
SS	AdPart-NA	2,743	120	320	3,203	1	1	40	75	3,920
	TriAD	6,023	1,519	2,387	17,586	6	4	114	369	912
	TriAD-SG	5,392	1,774	4,636	21,567	9	5	10	333	755
	Urika-GD	5,835	2,396	1,871	6,951	1,442	720	1,588	2,259	1,211
	H-RDF-3X	7,004	2,640	7,957	7,175	1,635	1,586	1,965	3,412	841
	H-RDF-3X (in-memory)	6,841	2,597	7,948	7,551	1,596	1,594	1,926	3,397	839
	SHAPE	25,319	4,387	25,360	15,026	1,603	1,574	1,567	5,575	337
	DREAM	13,031,410	98,263	2,358	4,700,381	18	14	10,755	12,110	1
	DREAM (cached)	1,843,376	98,263	<1	83,053	18	14	468	911	12

- Specialized for Graphs
- 1000x of CPU cores, TBs of RAM
- NUMA architecture
    - Global memory address space
    - Specialized network
    - Transparent to programmer

Spartex: RDF meets Graph Analytics

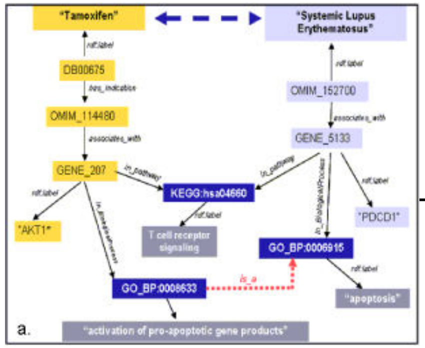
- *In: IEEE Trans. On Parallel and Distributed Systems, 2017*

RDF Analytics: Drug Repositioning

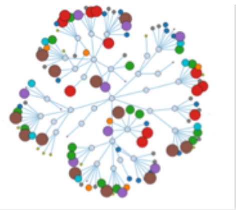
Disease SPARQL Query

RDF data

10g ORACLE DATABASE



Graph Engine



Betweenness, Closeness

Degree Centrality

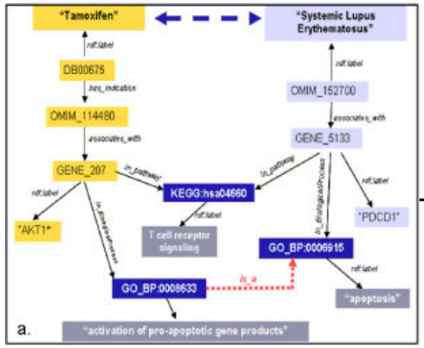
Ranked Drug Candidates

RDF Analytics: Drug Repositioning

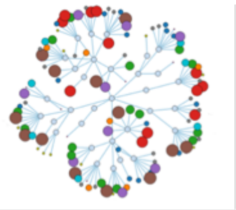
RDF data

Disease SPARQL Query

10g ORACLE DATABASE



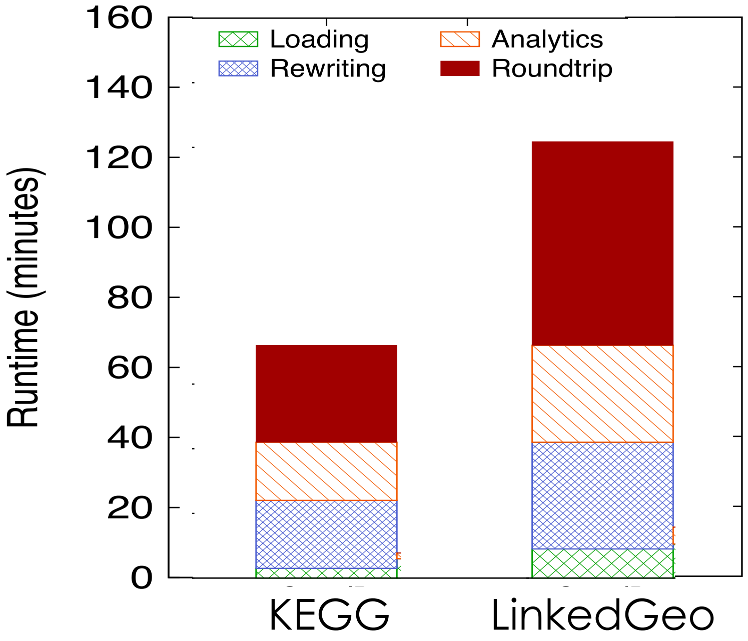
Graph Engine



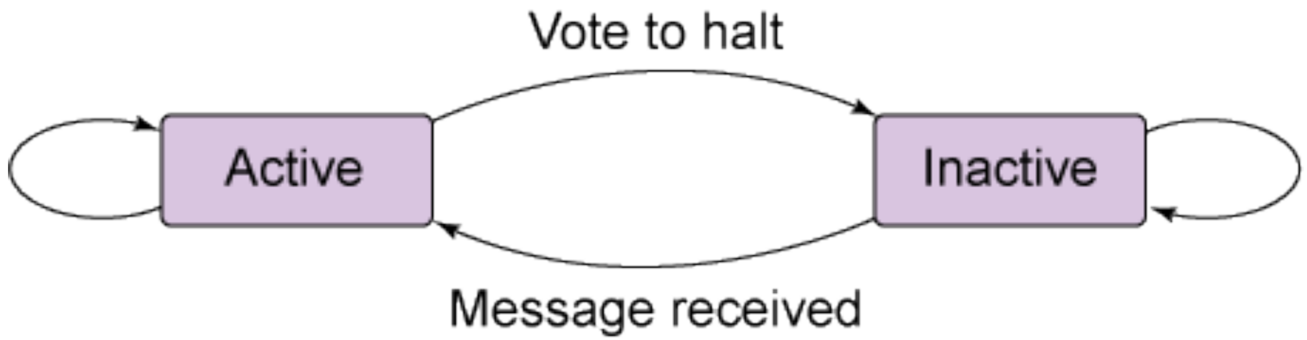
Betweenness, Closeness

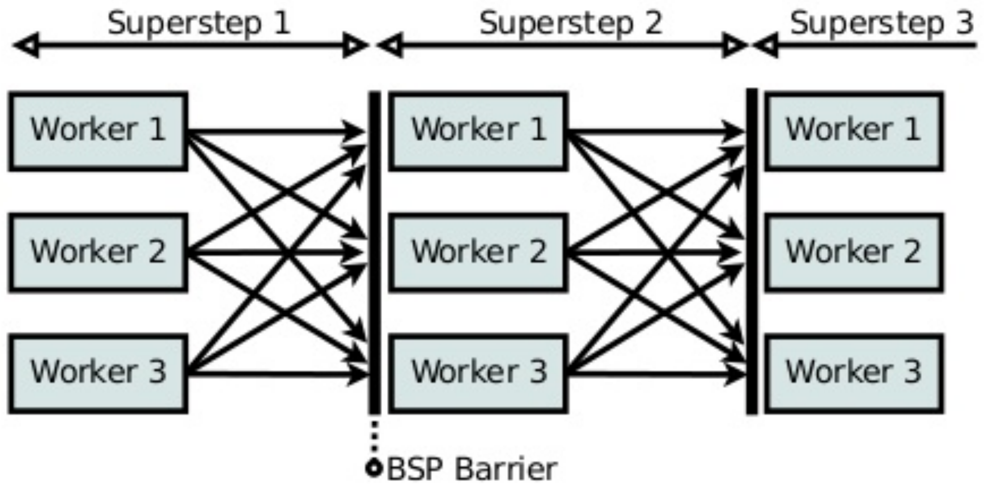
Degree Centrality

Ranked Drug Candidates

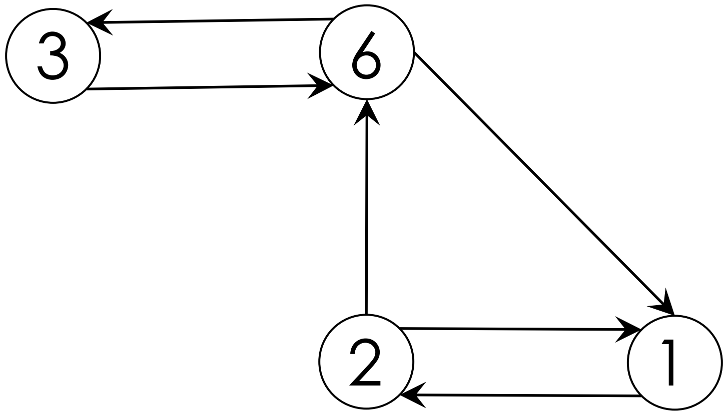


Vertex-Centric Framework: Pregel, …



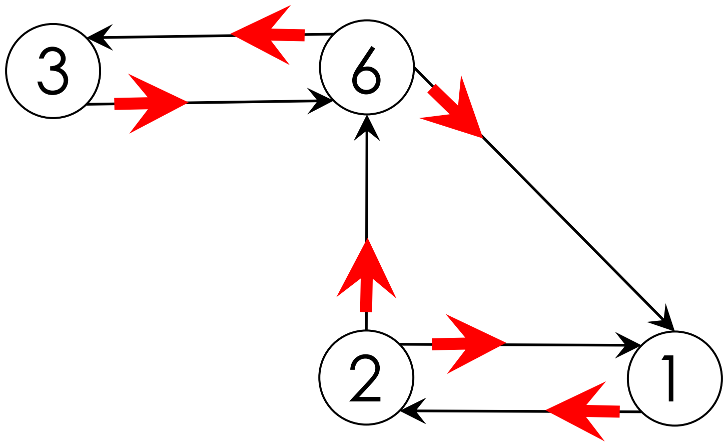


Pregel Example: **MAX**



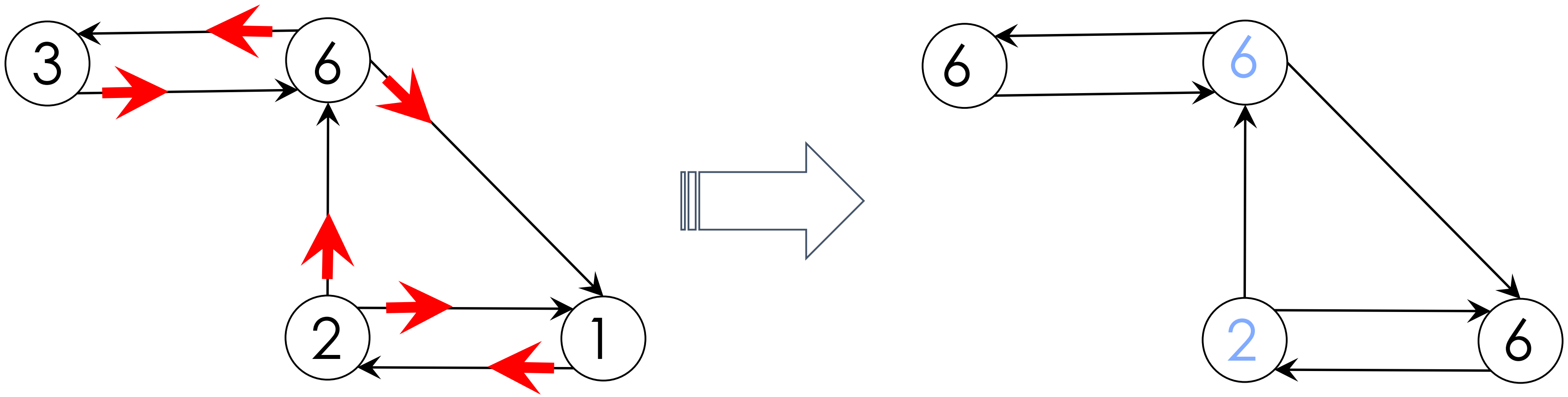
Example from [Malewich et al., SIGMOD, 2010]

Pregel Example: **MAX**



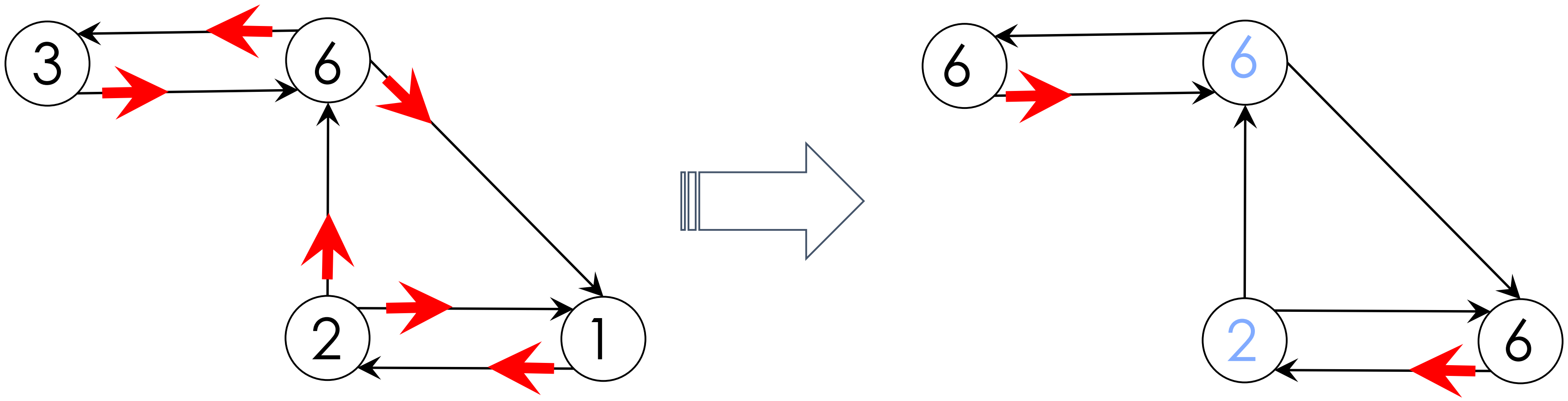
Example from [Malewich et al., SIGMOD, 2010]

Pregel Example: **MAX**



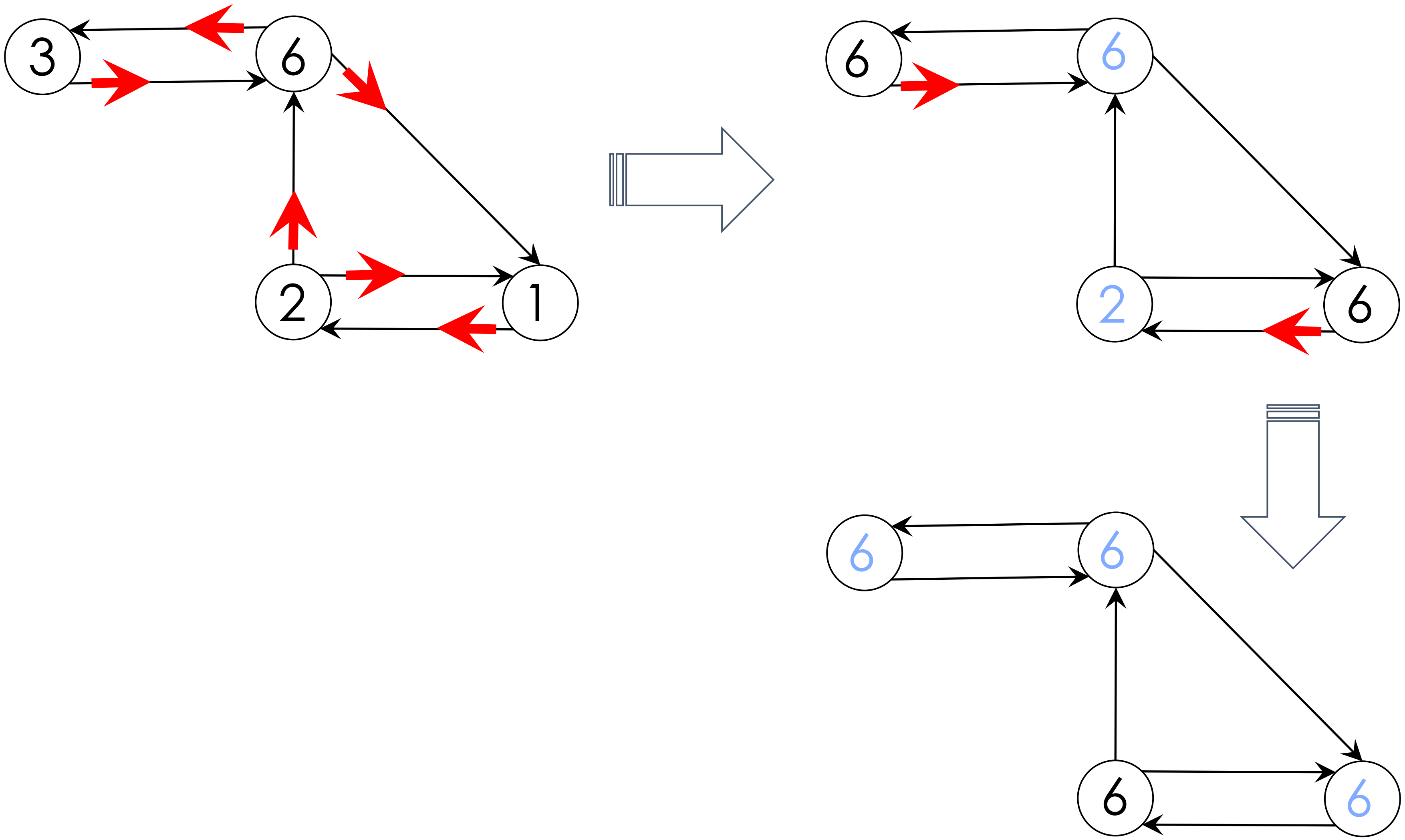
Example from [Malewich et al., SIGMOD, 2010]

Pregel Example: **MAX**



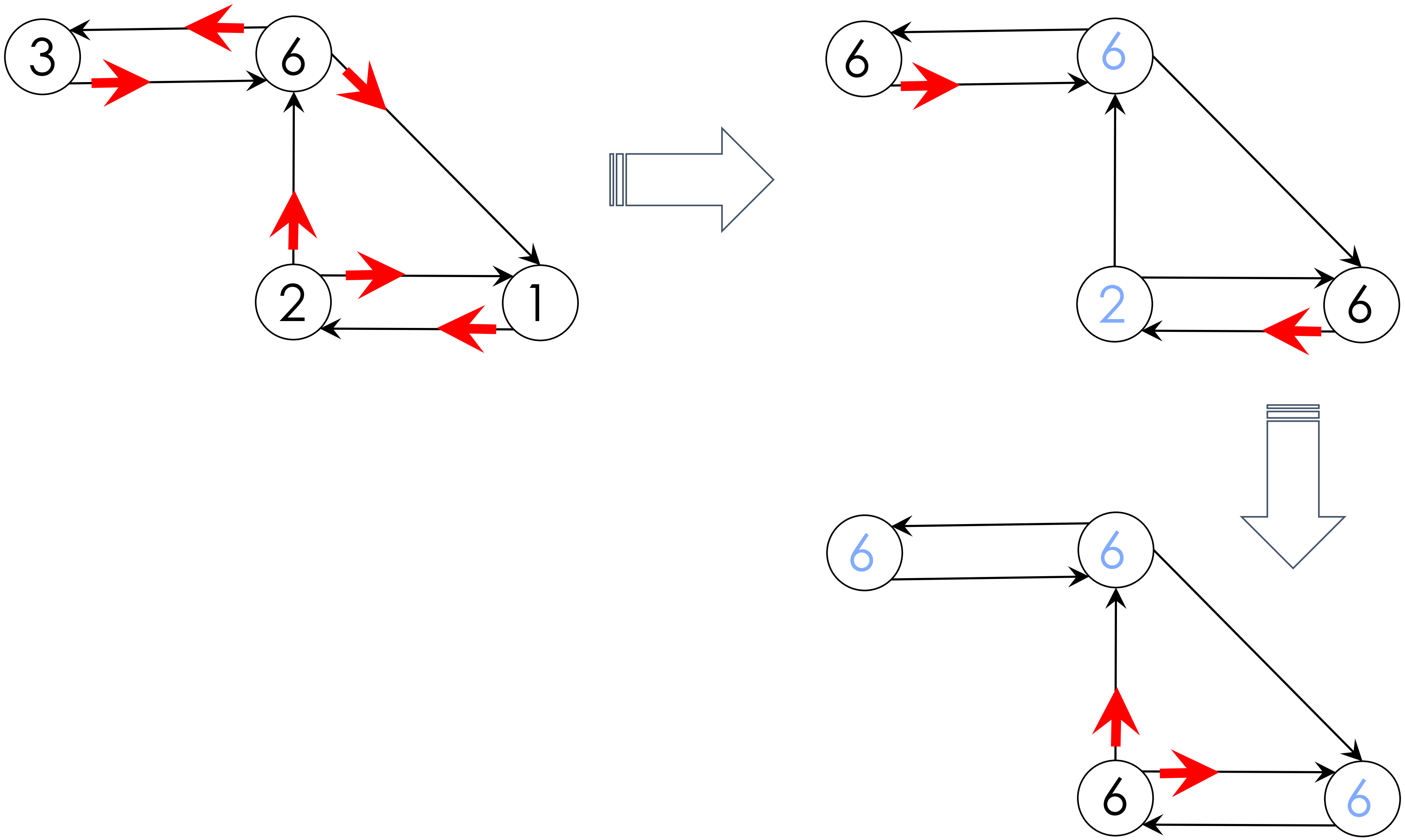
Example from [Malewich et al., SIGMOD, 2010]

Pregel Example: **MAX**



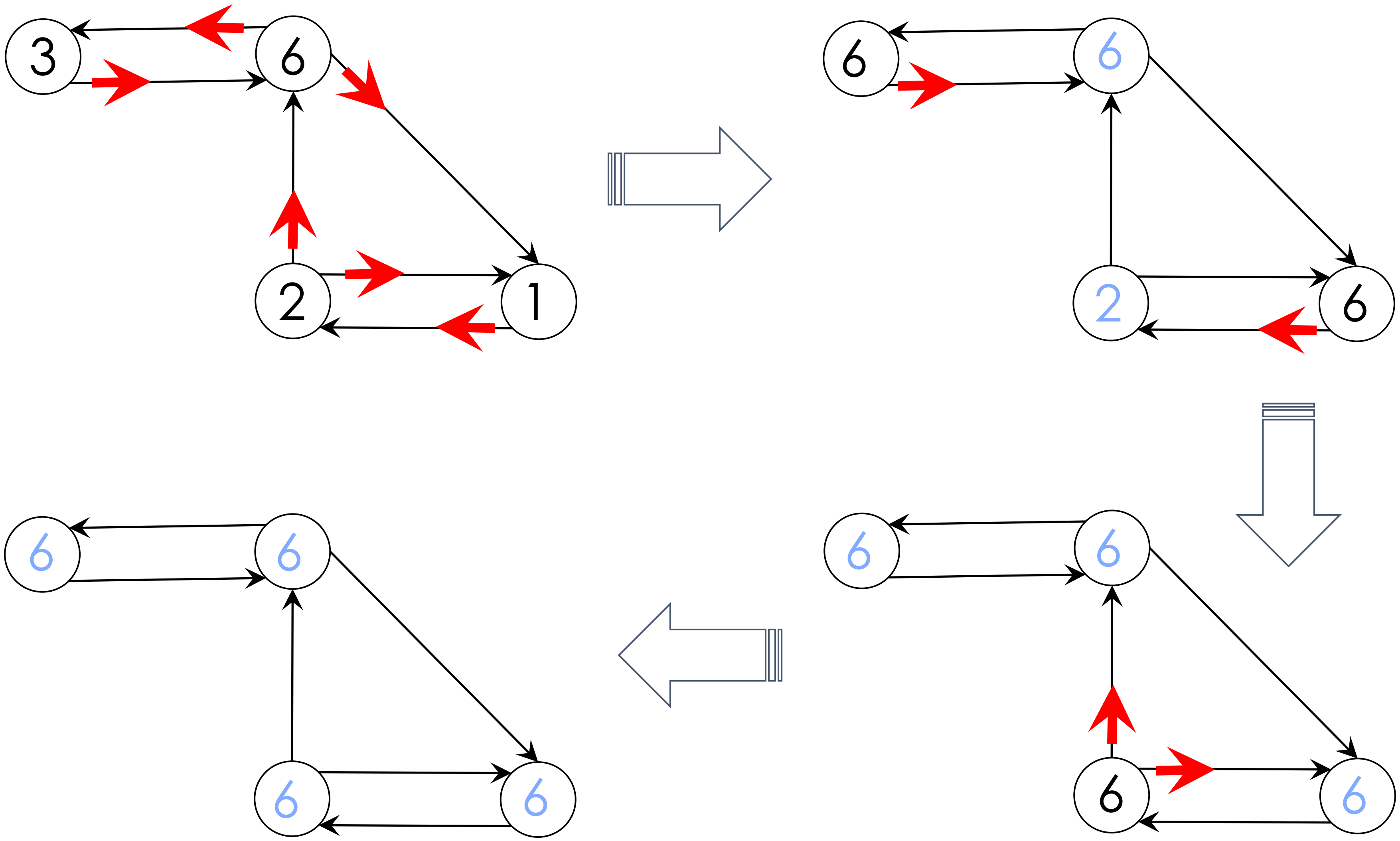
Example from [Malewich et al., SIGMOD, 2010]

Pregel Example: **MAX**



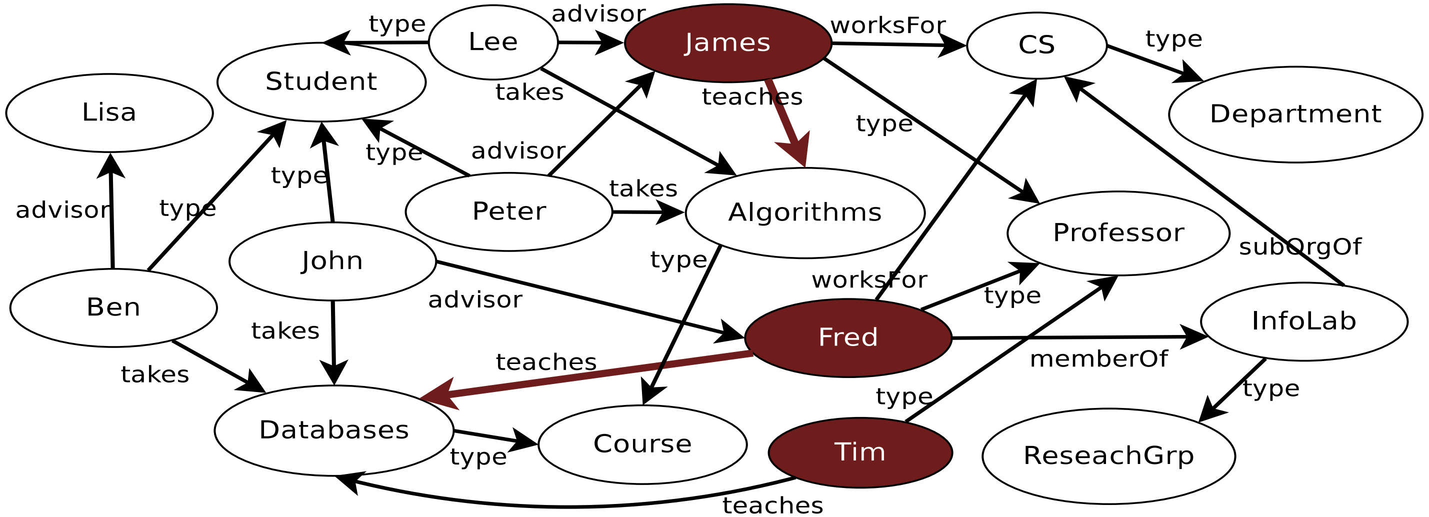
Example from [Malewich et al., SIGMOD, 2010]

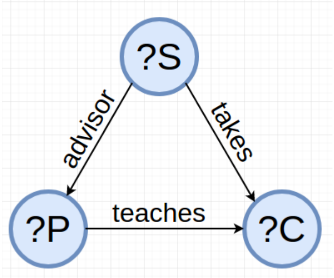
Pregel Example: **MAX**

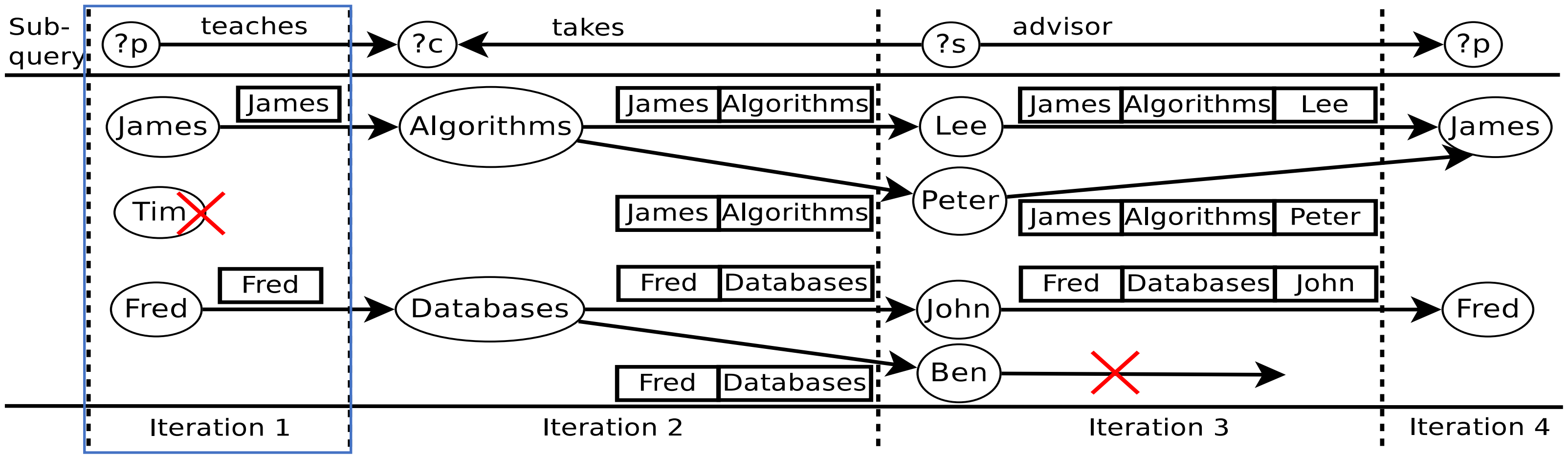


Example from [Malewich et al., SIGMOD, 2010]

SPARTEX: RDF @ Vertex-centric
[Kalnis et al., IEEE-TPDS, 2017]

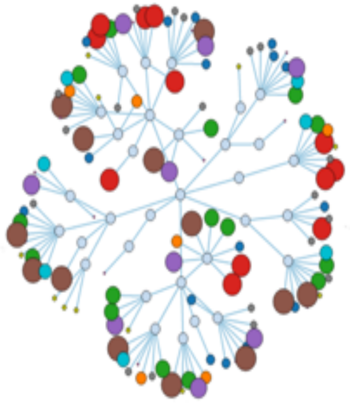






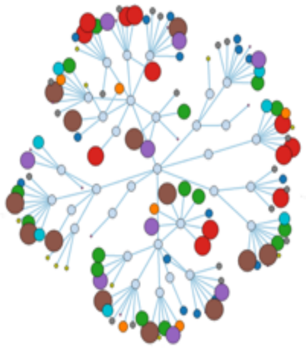
SPARTEX syntax (1)

```
PREFIX sptx: <http://www.spartex.com/analytics/>
CALL com.sptx.algo.centrality() AS sptx:centrality
CALL com.sptx.algo.PageRank(max_iter) AS sptx:pRank
SELECT ?s WHERE {
    ?p   teaches ?c .
    ?s   takes  ?c .
    ?s   advisor ?p .
    ?p   sptx:pRank     ?rank .
    ?c   sptx:centrality ?cent .
    FILTER (?rank > val1 && ?cent > val2)
}
```

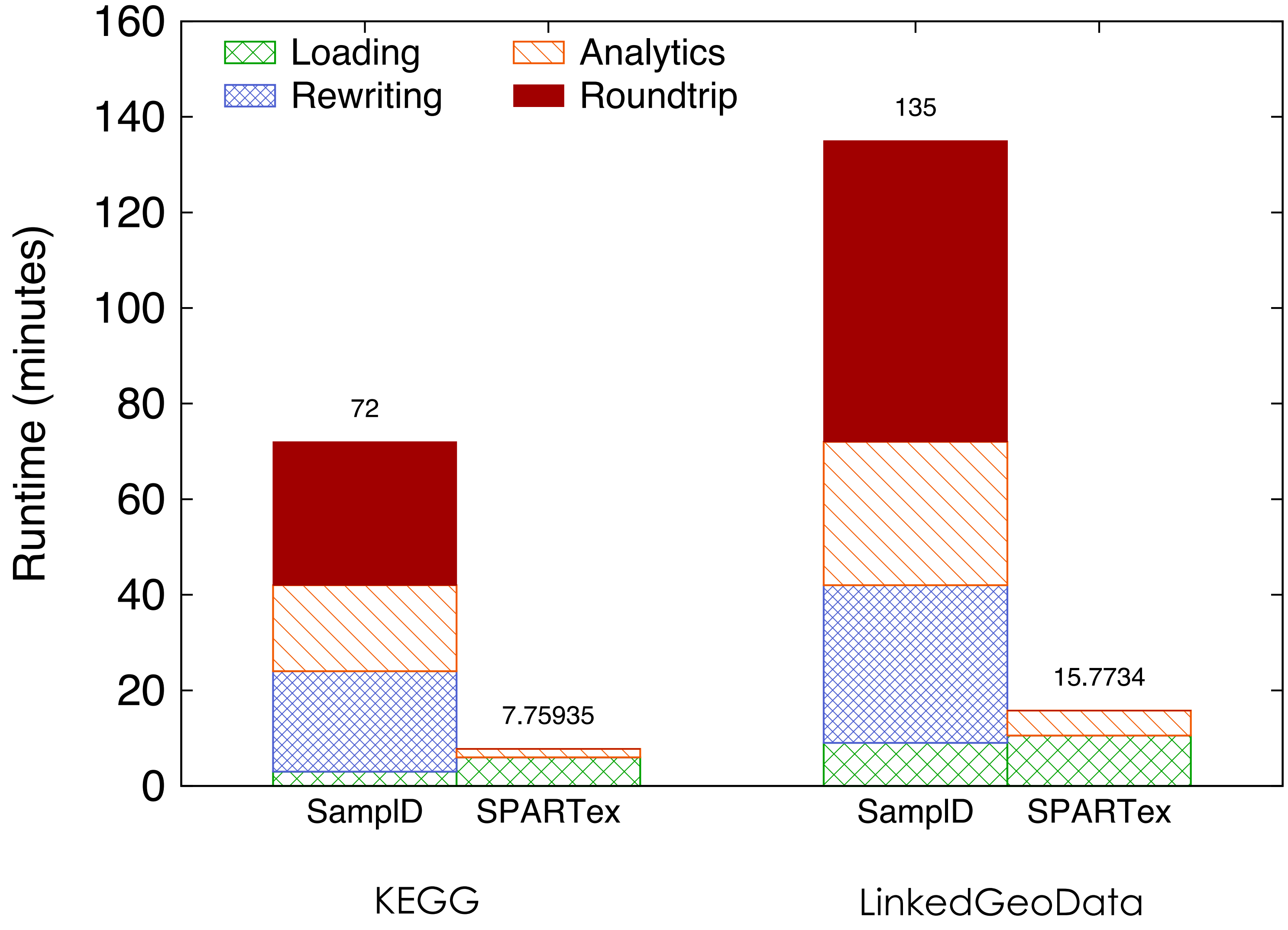


SPARTEX syntax (2)

```
PREFIX sptx: <http://www.spartex.com/analytics/>
CALL com.sptx.algo.centrality() AS sptx:centrality
CALL com.sptx.algo.PageRank(max_iter) AS sptx:pRank
ADD TRIPLE {?p sptx:popular "T" . } WHERE {
    ?p   teaches ?c .
    ?s   takes  ?c .
    ?s   advisor ?p .
    ?p   sptx:pRank    ?rank .
    ?c   sptx:centrality ?cent .
    FILTER (?rank > val1 && ?cent > val2)
}
FILTER_VERTEX AS start WHERE {
  ?p sptx:popular "T" .
}
CALL algo:SSSP() USING start AS sptx:sssp
```



SPARTEX: 10x faster

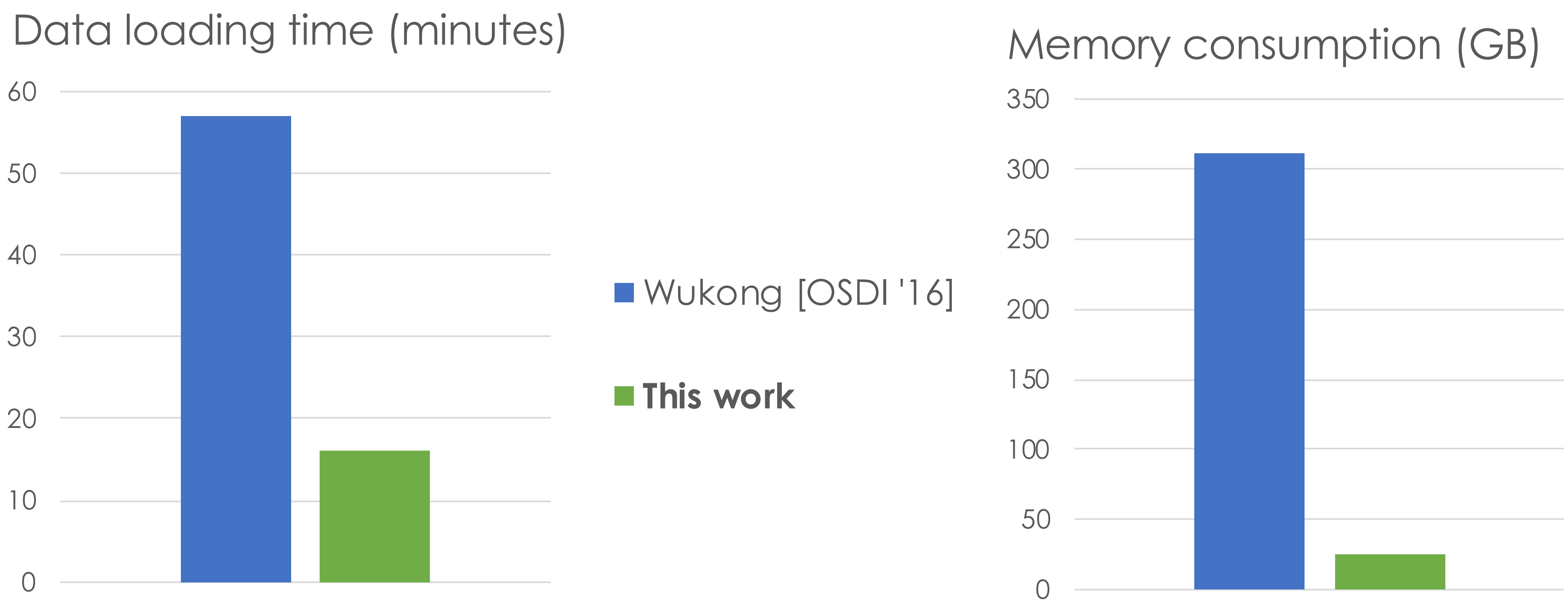


MAGiQ: Portability and Scalability

- *Demo in PVLDB, 2018*
- *In: Proc. of EuroSys, 2019*

Existing Engines and Large Graphs

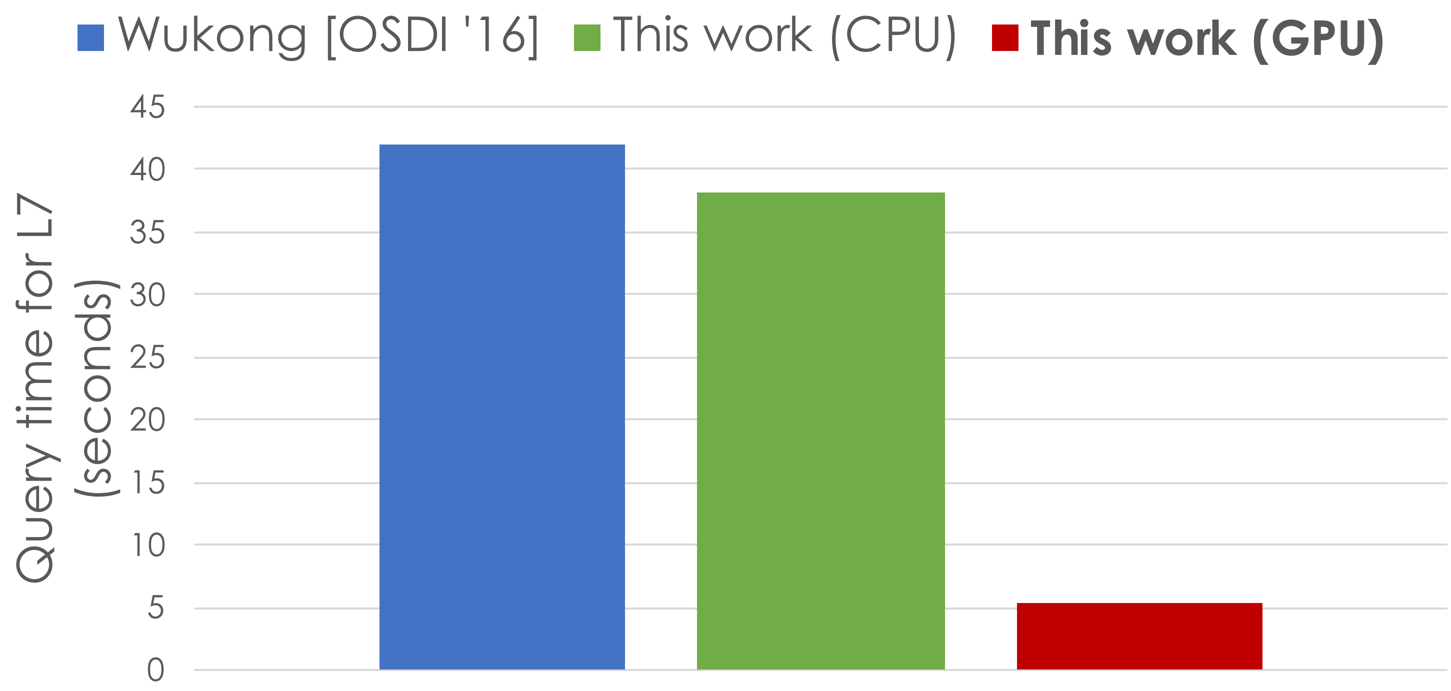
- Expensive indices [almost no indices]



Data: LUBM-1B (1.3B edges); *Hardware*: Linux server, 24 cores@2.4Ghz, 512GB

Existing Engines and Data-intensive Queries

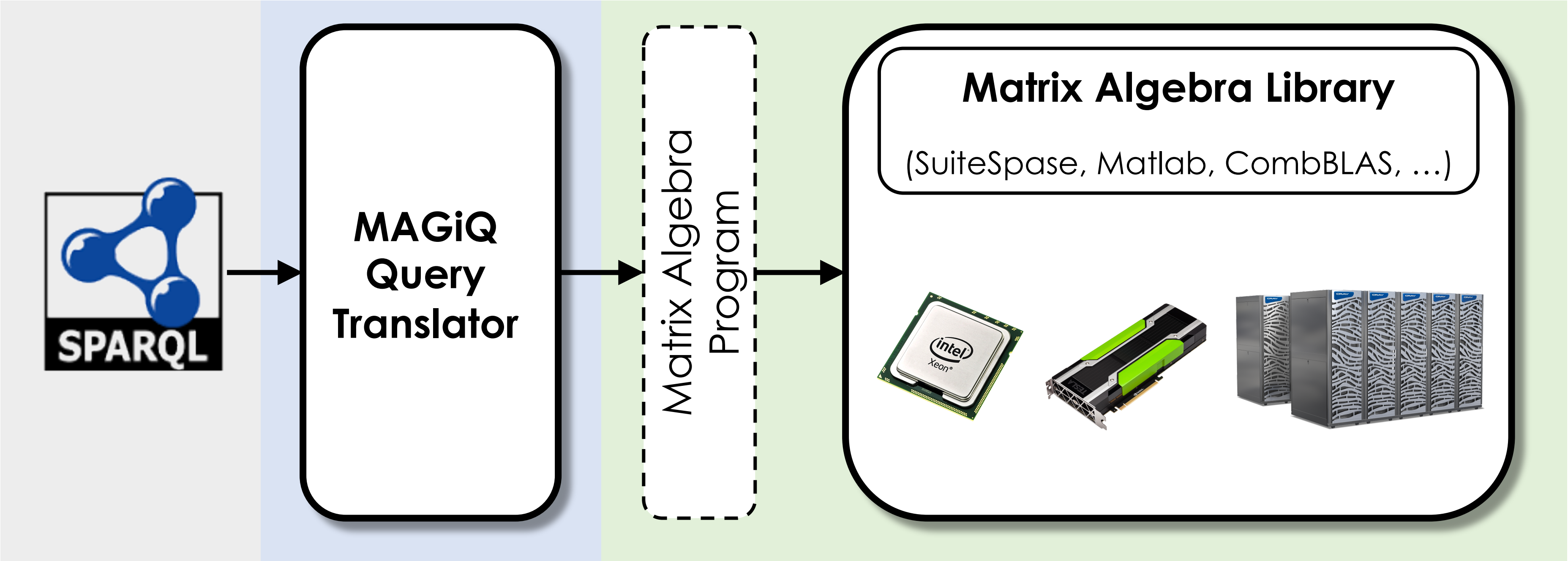
- Slow and difficult to port to different HW [effortlessly portable]
  - Distributed – AdPart [VLDBJ '16]
  - GPU – Wukong+G [ATC '18]



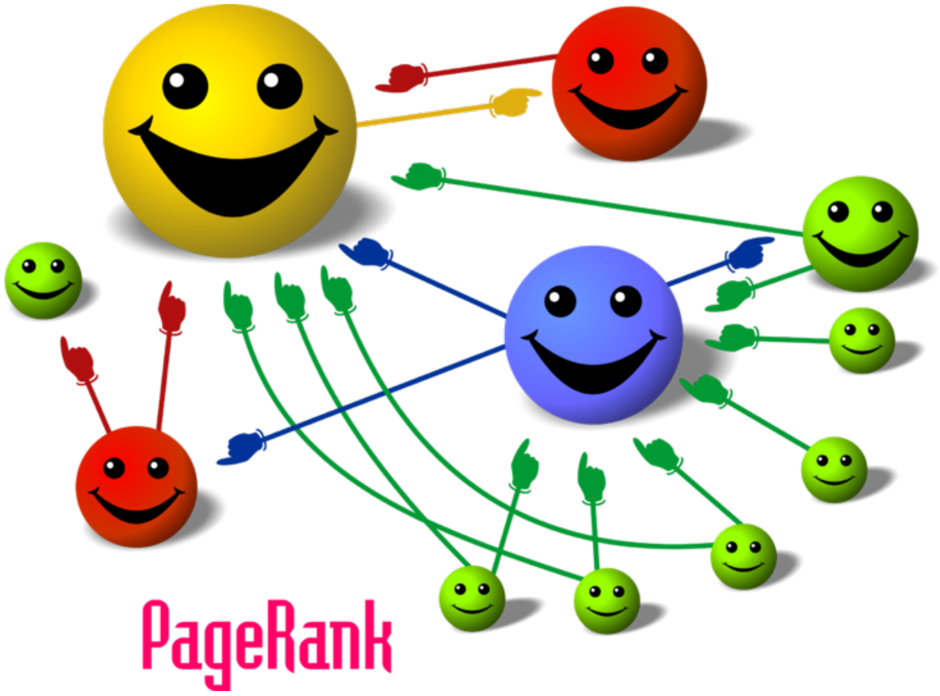
Data: LUBM-1B (1.3B edges); *Hardware*: Linux server, 24 cores@2.4Ghz, 512GB, NVIDIA Quadro P6000

Our Proposal: **MAGiQ**

- **Translate graph queries into matrix algebra programs**:
  - Decouple query evaluation logic from particular HW [**Portability**]
  - Compact sparse matrix representation of input graphs [**Scalability**]
  - Utilize highly efficient existing matrix algebra libraries [**Efficiency**]

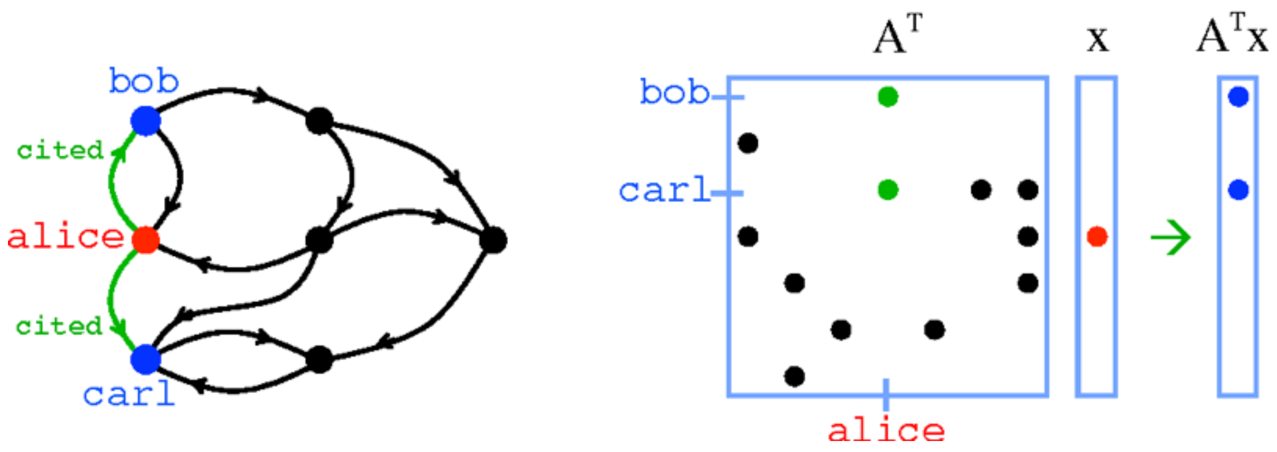


Graphs as Matrices… old news?



$$\mathbf{R} = (\mathbf{I} - d\mathcal{M})^{-1}\frac{1-d}{N}\mathbf{1}$$

GraphBLAS

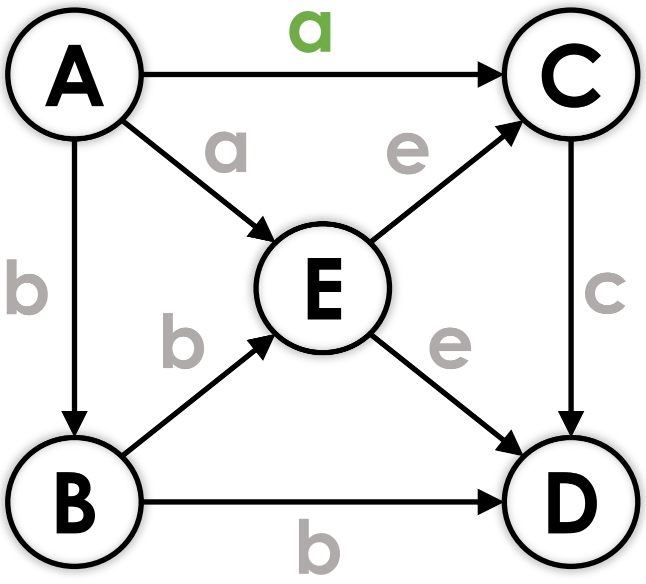


- Represent graphs as sparse matrices
- Define common operations
- Matrix x Vector → BFS

Algorithm (Problem)	Canonical Complexity	LA-Based Complexity
Breadth-first search	$\Theta(m)$	$\Theta(m)$
Betweenness Centrality (unweighted)	$\Theta(mn)$	$\Theta(mn)$
All-pairs shortest-paths (dense)	$\Theta(n^3)$	$\Theta(n^3)$
Prim (MST)	$\Theta(m+n \log n)$	$\Theta(n^2)$
Borůvka (MST)	$\Theta(m \log n)$	$\Theta(m \log n)$
Edmonds-Karp (Max Flow)	$\Theta(m^2 n)$	$\Theta(m^2 n)$
Greedy MIS (MIS)	$\Theta(m+n \log n)$	$\Theta(mn+n^2)$
Luby (MIS)	$\Theta(m+n \log n)$	$\Theta(m \log n)$

MAGiQ – RDF Graph Representation

G



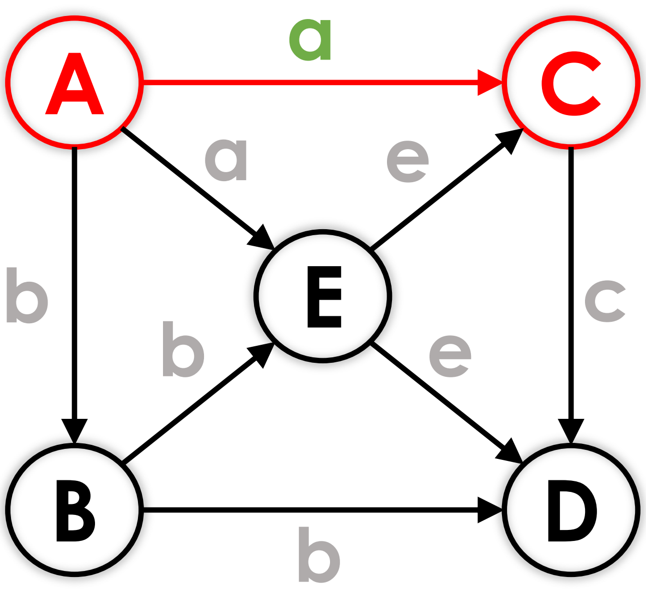
A

	A	B	C	D	E
A		b	a		a
B				b	b
C				c	
D					
E			e	e	

Pre-processing → Matrix operations → Post-processing

MAGiQ – RDF Graph Representation

G



A

	A	B	C	D	E
A		b	a		a
B				b	b
C				c	
D					
E			e	e	

Pre-processing → Matrix operations → Post-processing

Selection as Matrix Multiplication

$$\underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\mathbf{S}} \times \underbrace{\begin{pmatrix} a & b & c \\ a & b & c \\ a & b & c \end{pmatrix}}_{\mathbf{M}} = \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ a & b & c \\ 0 & 0 & 0 \end{pmatrix}}_{\mathbf{C}}$$

- S: 1 at row i → select row i from M

Generalized Matrix Selection

$$\underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\mathbf{S}*b} \otimes \underbrace{\begin{pmatrix} a & b & c \\ a & b & c \\ a & b & c \end{pmatrix}}_{\mathbf{M}} = \underbrace{\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{C}$$

- Semi-ring with
  - `isEqual` instead of multiplication
  - `OR` instead of addition

MAGiQ – Algebraic Operations

Matrix-matrix multiplication over `isEqual , OR` **semi-ring**

$$
\begin{array}{c}
\textbf{S} \\
\begin{array}{c|ccccc|}
 & A & B & C & D & E \\
\hline
A & & & & & \\
B & & & & & \\
C & & & e & & \\
D & & & & & \\
E & & & & & e \\
\hline
\end{array}
\end{array}
\otimes
\begin{array}{c}
\textbf{A} \\
\begin{array}{c|ccccc|}
 & A & B & C & D & E \\
\hline
A & & b & a & & a \\
B & & & & b & b \\
C & & & & c & \\
D & & & & & \\
E & & & e & e & \\
\hline
\end{array}
\end{array}
=
\begin{array}{c}
\\
\begin{array}{c|ccccc|}
 & A & B & C & D & E \\
\hline
A & & & & & \\
B & & & & & \\
C & & & & & \\
D & & & & & \\
E & & & 1 & 1 & \\
\hline
\end{array}
\end{array}
$$

MAGiQ – Algebraic Operations

Matrix-matrix multiplication over **`isEqual , OR`** **semi-ring**

S

	A	B	C	D	E
A					
B					
C			e		
D					
E					e

$\otimes$

A

	A	B	C	D	E
A		b	a		a
B				b	b
C				c	
D					
E			e	e	

$=$

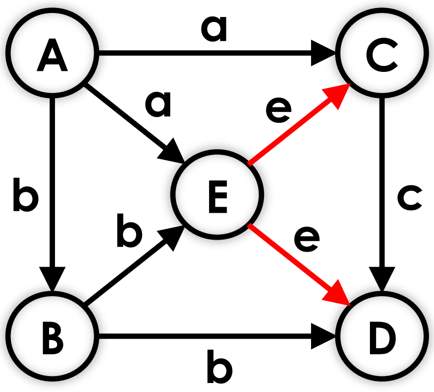
	A	B	C	D	E
A					
B					
C					
D					
E			1	1	

MAGiQ – Algebraic Operations

Matrix-matrix multiplication over `isEqual , OR` **semi-ring**

Row selection with predicate:

neighbors connected to **C** and **E** with **e** outgoing edges



S

	A	B	C	D	E
A					
B					
C			e		
D					
E					e

$\otimes$

A

	A	B	C	D	E
A		b	a		a
B				b	b
C				c	
D					
E			e	e	

=

	A	B	C	D	E
A					
B					
C					
D					
E			1	1	

MAGiQ – Algebraic Operations

`any`(**M**): reduction with **`OR`**

$$\texttt{any}\left(\begin{array}{|c:c:c:c:c|}\hline & 1 & & & 1 \\ \hdashline & & & & \\ \hdashline & & & & \\ \hdashline & 1 & & & \\ \hdashline & & & & \\ \hline\end{array}\right) = \begin{array}{|c|}\hline 1 \\ \hdashline \\ \hdashline \\ \hdashline 1 \\ \hdashline \\ \hline\end{array}$$

MAGiQ – Algebraic Operations

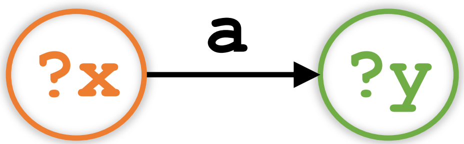
`diag(v)`: construct diagonal selection matrix

$$\mathtt{diag}\left(\begin{bmatrix} 1 \\ \ \\ \ \\ 1 \\ \ \end{bmatrix}\right) = \begin{bmatrix} 1 & & & & \\ & & & & \\ & & & & \\ & & & 1 & \\ & & & & \end{bmatrix}$$

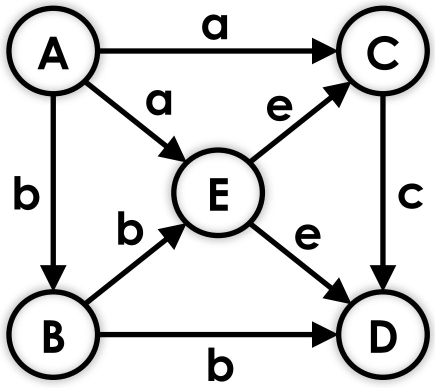
MAGiQ – Query Translation

Single edge translation

```
SELECT ?x ?y WHERE {
   ?x    <a>   ?y  .
}
```



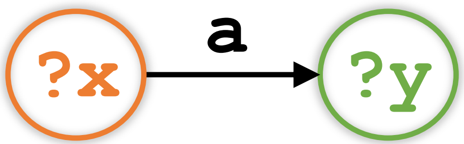
$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$



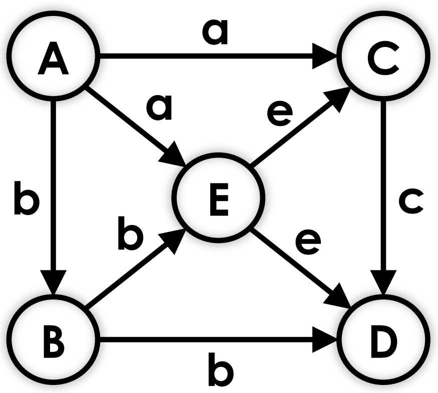
MAGiQ – Query Translation

Single edge translation

```
SELECT ?x ?y WHERE {
  ?x   <a>   ?y .
}
```



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$



$\mathbf{I} * a$

	A	B	C	D	E
A	a				
B		a			
C			a		
D				a	
E					a

$\otimes$

$\mathbf{A}$

	A	B	C	D	E
A		b	a		a
B				b	b
C				c	
D					
E			e	e	

$=$

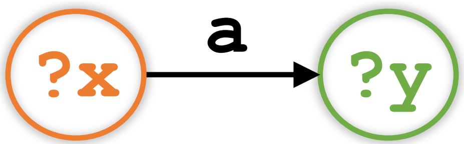
$\mathbf{M}_{xy}$

	A	B	C	D	E
A			1		1
B					
C					
D					
E					

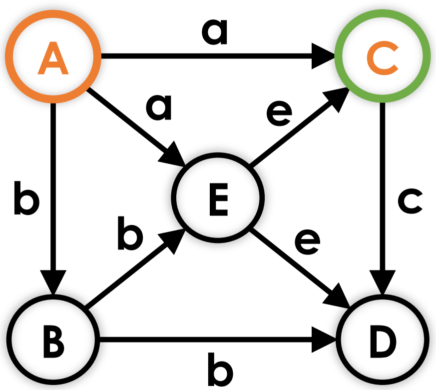
MAGiQ – Query Translation

Single edge translation

```
SELECT ?x ?y WHERE {
   ?x   <a>   ?y .
}
```



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$



$\mathbf{I} * a$

	A	B	C	D	E
A	a				
B		a			
C			a		
D				a	
E					a

$\otimes$

$\mathbf{A}$

	A	B	C	D	E
A		b	a		a
B				b	b
C				c	
D					
E			e	e	

=

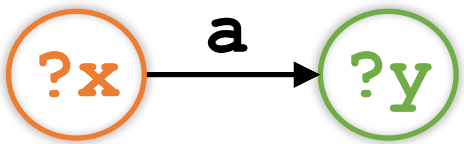
$\mathbf{M}_{xy}$

	A	B	C	D	E
A			$\boxed{1}$		1
B					
C					
D					
E					

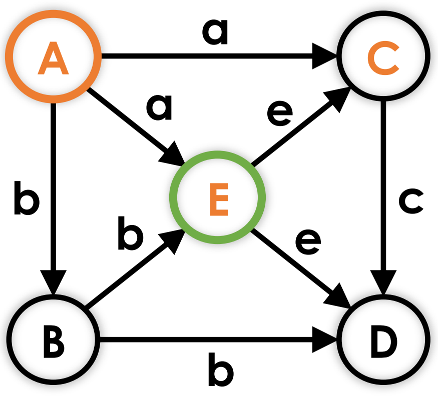
MAGiQ – Query Translation

Single edge translation

```
SELECT ?x ?y WHERE {
  ?x   <a>  ?y .
}
```



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$



$\mathbf{I} * a$

	A	B	C	D	E
A	a				
B		a			
C			a		
D				a	
E					a

$\otimes$

$\mathbf{A}$

	A	B	C	D	E
A		b	a		a
B				b	b
C				c	
D					
E			e	e	

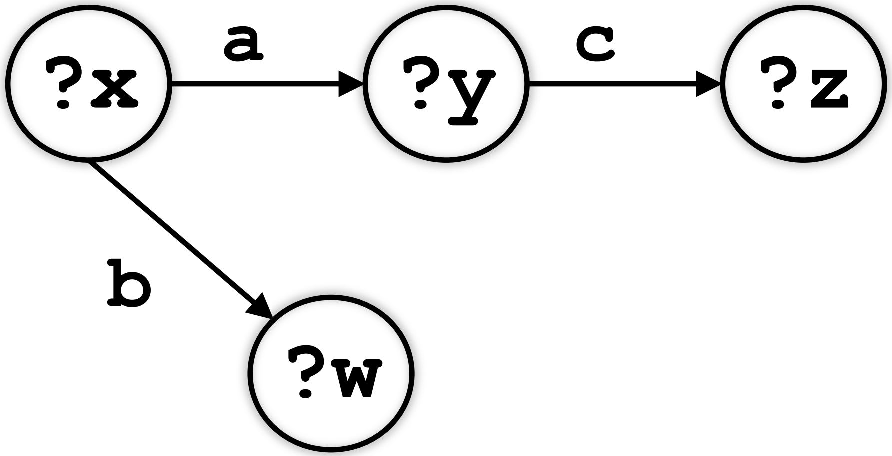
=

$\mathbf{M}_{xy}$

	A	B	C	D	E
A			1		$\boxed{1}$
B					
C					
D					
E					

MAGiQ – Query Translation

Graph query translation



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$

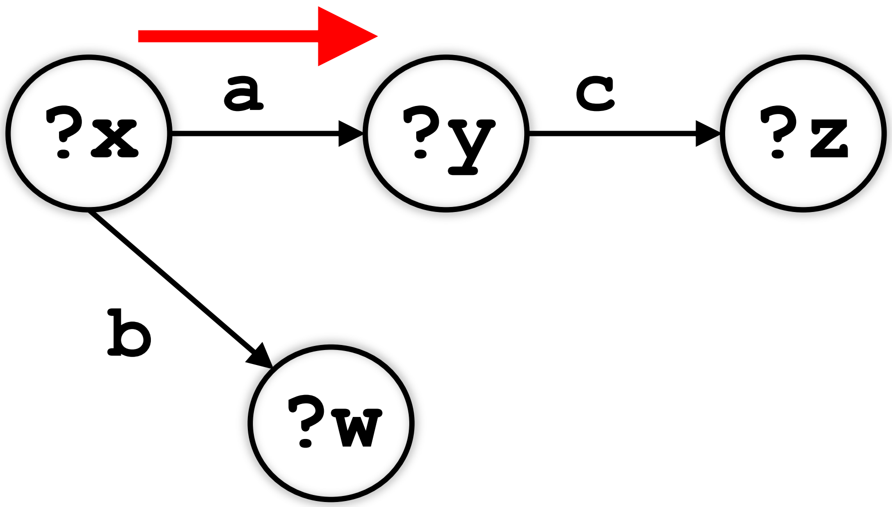
$$\mathbf{M}_{yz} = \mathtt{diag}(\mathtt{any}(\mathbf{M}'_{xy})) * c \otimes \mathbf{A}$$

$$\mathbf{M}_{xy} = \mathbf{M}_{xy} \times \mathtt{diag}(\mathtt{any}(\mathbf{M}_{yz}))$$

$$\mathbf{M}_{xw} = \mathtt{diag}(\mathtt{any}(\mathbf{M}_{xy})) * b \otimes \mathbf{A}$$

MAGiQ – Query Translation

Graph query translation



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$

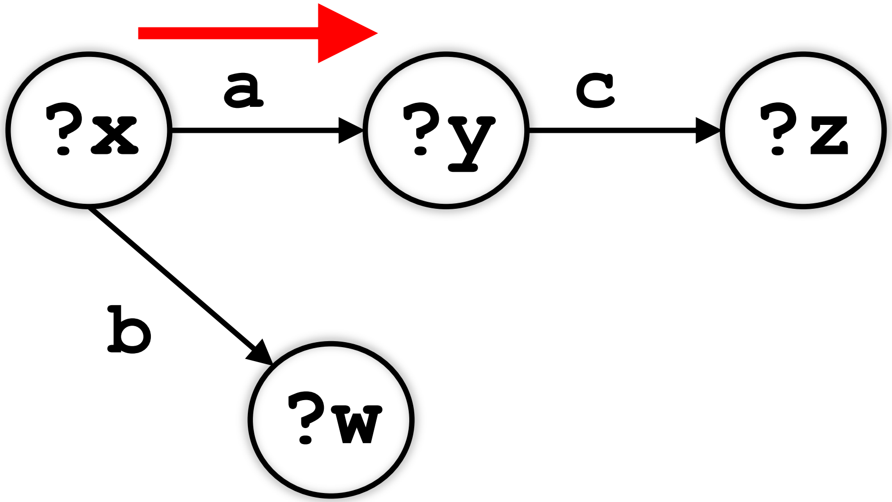
$$\mathbf{M}_{yz} = \mathtt{diag}(\mathtt{any}(\mathbf{M}'_{xy})) * c \otimes \mathbf{A}$$

$$\mathbf{M}_{xy} = \mathbf{M}_{xy} \times \mathtt{diag}(\mathtt{any}(\mathbf{M}_{yz}))$$

$$\mathbf{M}_{xw} = \mathtt{diag}(\mathtt{any}(\mathbf{M}_{xy})) * b \otimes \mathbf{A}$$

MAGiQ – Query Translation

Graph query translation



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$

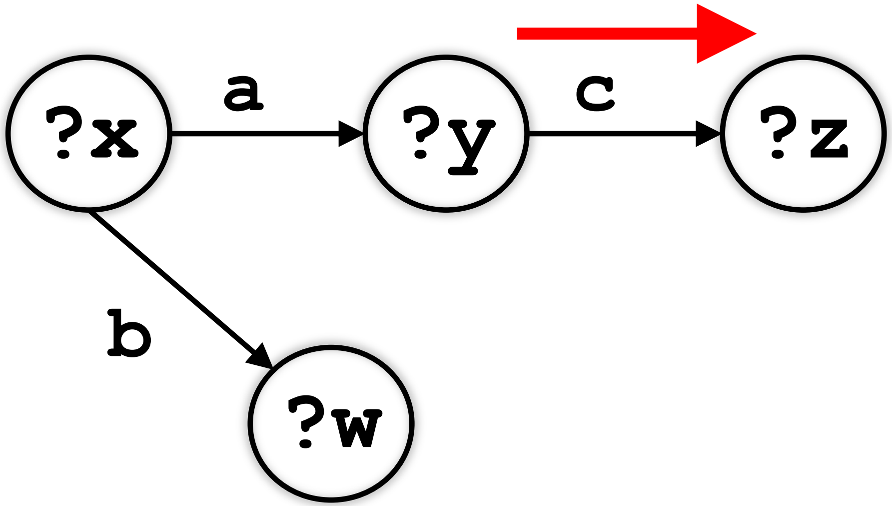
$$\mathbf{M}_{yz} = \mathtt{diag}(\mathtt{any}(\mathbf{M}'_{xy})) * c \otimes \mathbf{A}$$

$$\mathbf{M}_{xy} = \mathbf{M}_{xy} \times \mathtt{diag}(\mathtt{any}(\mathbf{M}_{yz}))$$

$$\mathbf{M}_{xw} = \mathtt{diag}(\mathtt{any}(\mathbf{M}_{xy})) * b \otimes \mathbf{A}$$

MAGiQ – Query Translation

Graph query translation



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$

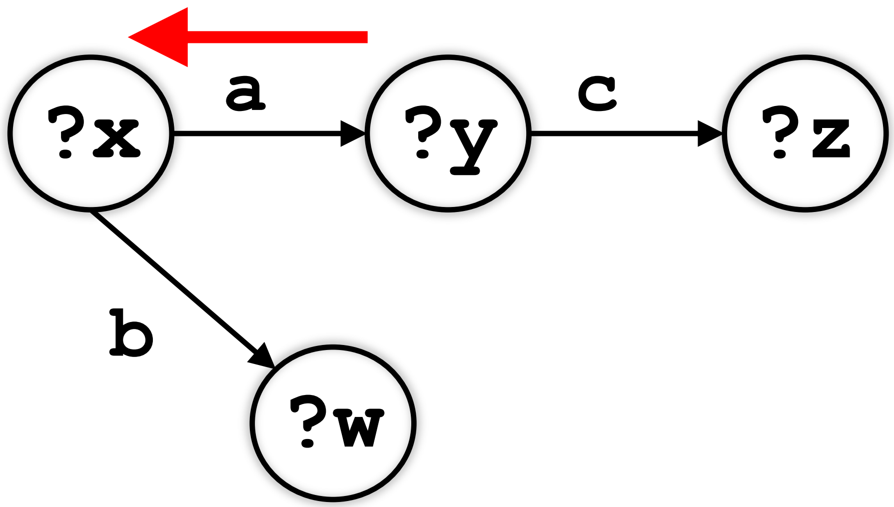
$$\mathbf{M}_{yz} = \mathtt{diag}(\mathtt{any}(\mathbf{M}'_{xy})) * c \otimes \mathbf{A}$$

$$\mathbf{M}_{xy} = \mathbf{M}_{xy} \times \mathtt{diag}(\mathtt{any}(\mathbf{M}_{yz}))$$

$$\mathbf{M}_{xw} = \mathtt{diag}(\mathtt{any}(\mathbf{M}_{xy})) * b \otimes \mathbf{A}$$

MAGiQ – Query Translation

Graph query translation



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$

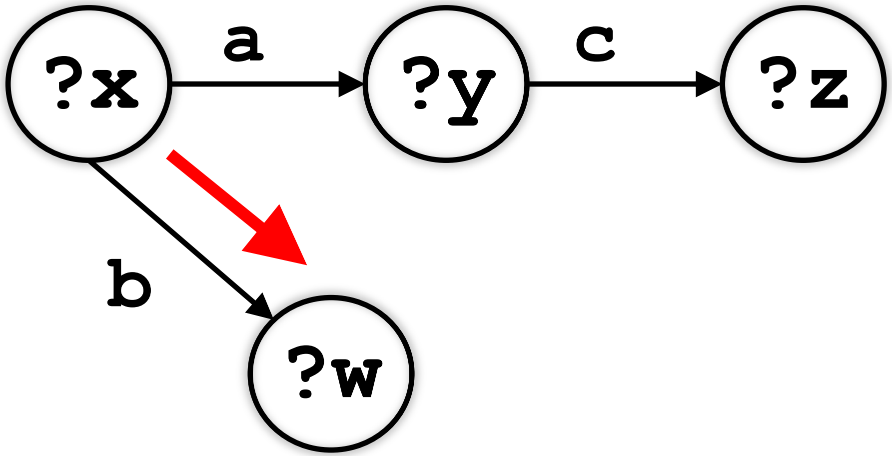
$$\mathbf{M}_{yz} = \mathtt{diag}(\mathtt{any}(\mathbf{M}'_{xy})) * c \otimes \mathbf{A}$$

$$\mathbf{M}_{xy} = \mathbf{M}_{xy} \times \mathtt{diag}(\mathtt{any}(\mathbf{M}_{yz}))$$

$$\mathbf{M}_{xw} = \mathtt{diag}(\mathtt{any}(\mathbf{M}_{xy})) * b \otimes \mathbf{A}$$

MAGiQ – Query Translation

Graph query translation



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$

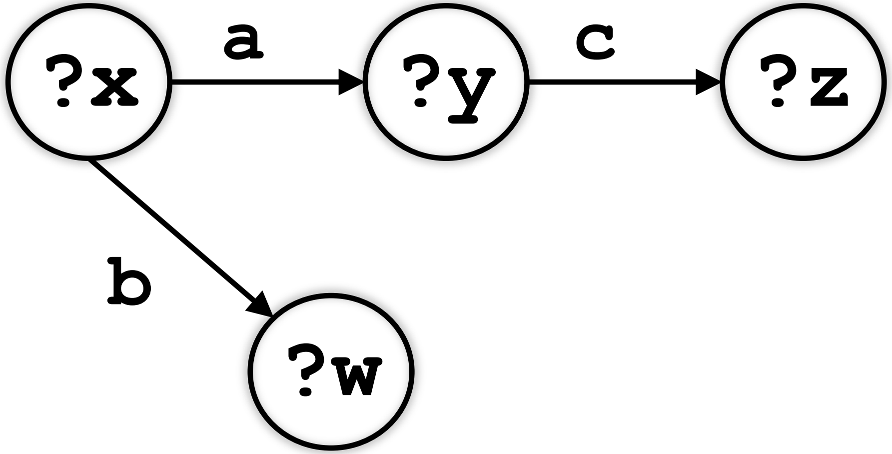
$$\mathbf{M}_{yz} = \mathtt{diag}(\mathtt{any}(\mathbf{M}'_{xy})) * c \otimes \mathbf{A}$$

$$\mathbf{M}_{xy} = \mathbf{M}_{xy} \times \mathtt{diag}(\mathtt{any}(\mathbf{M}_{yz}))$$

$$\mathbf{M}_{xw} = \mathtt{diag}(\mathtt{any}(\mathbf{M}_{xy})) * b \otimes \mathbf{A}$$

MAGiQ – Query Translation

Graph query translation



$$\mathbf{M}_{xy} = \mathbf{I} * a \otimes \mathbf{A}$$

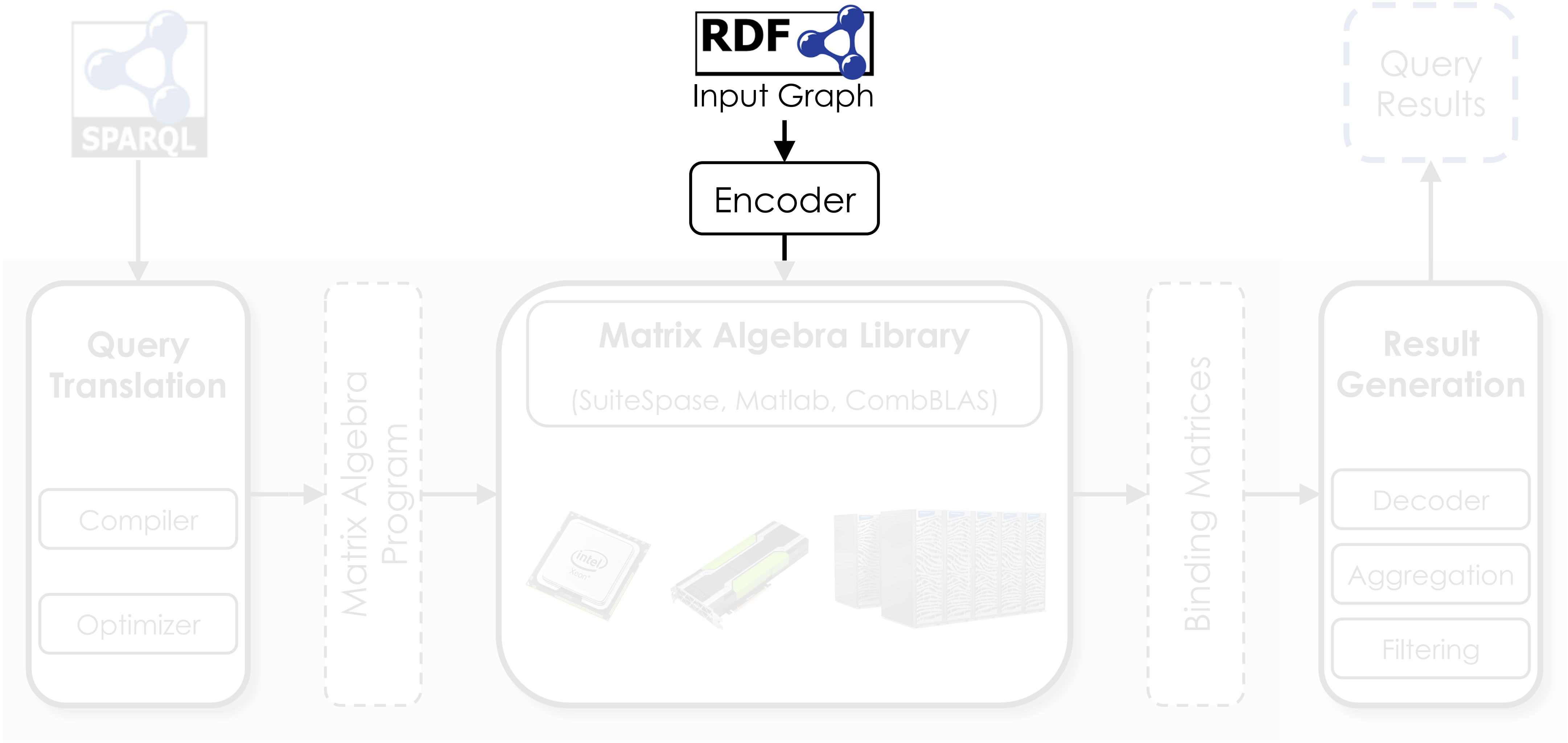
$$\mathbf{M}_{yz} = \mathtt{diag}(\mathtt{any}(\mathbf{M}'_{xy})) * c \otimes \mathbf{A}$$

$$\mathbf{M}_{xy} = \mathbf{M}_{xy} \times \mathtt{diag}(\mathtt{any}(\mathbf{M}_{yz}))$$

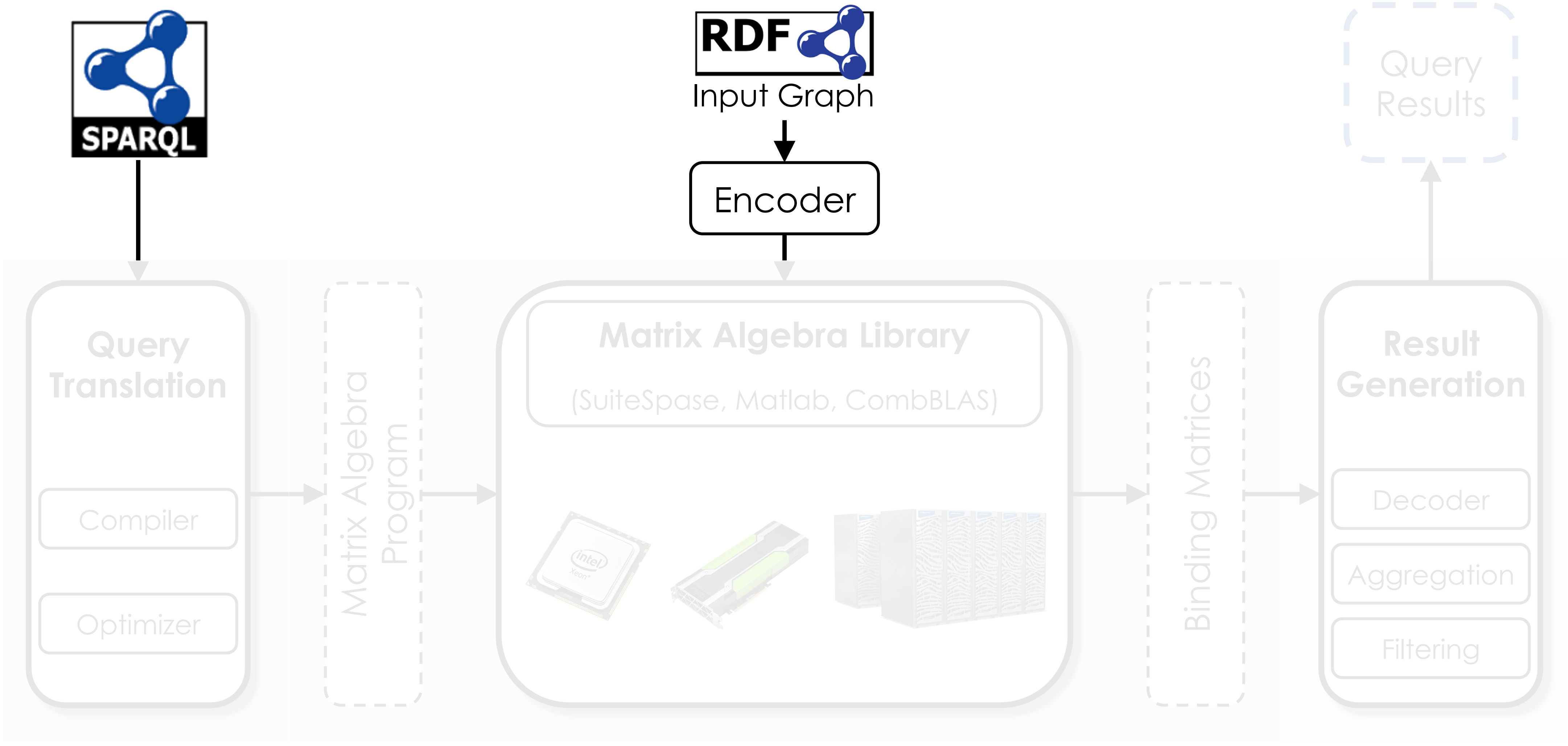
$$\mathbf{M}_{xw} = \mathtt{diag}(\mathtt{any}(\mathbf{M}_{xy})) * b \otimes \mathbf{A}$$

MAGiQ – Architecture

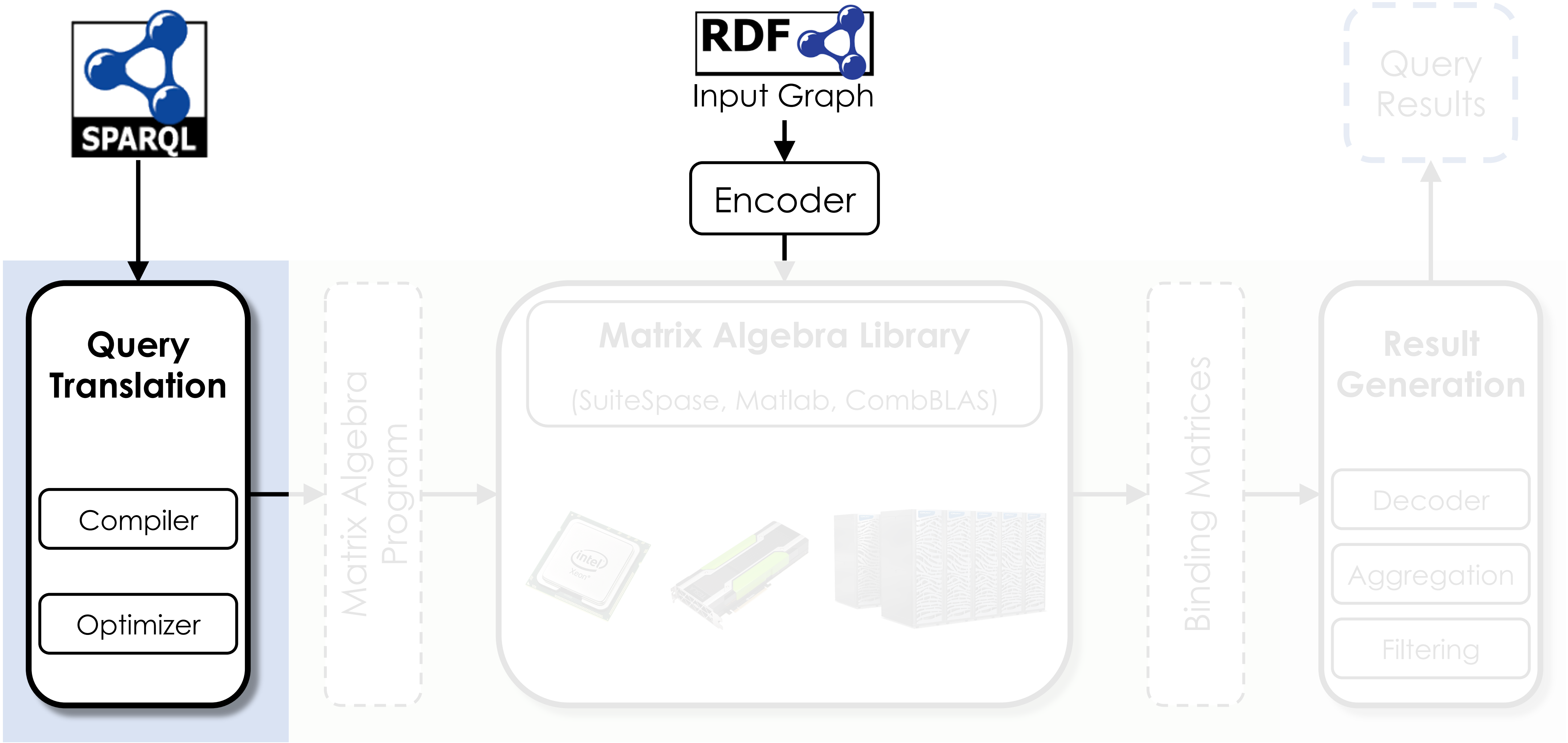
MAGiQ – Architecture



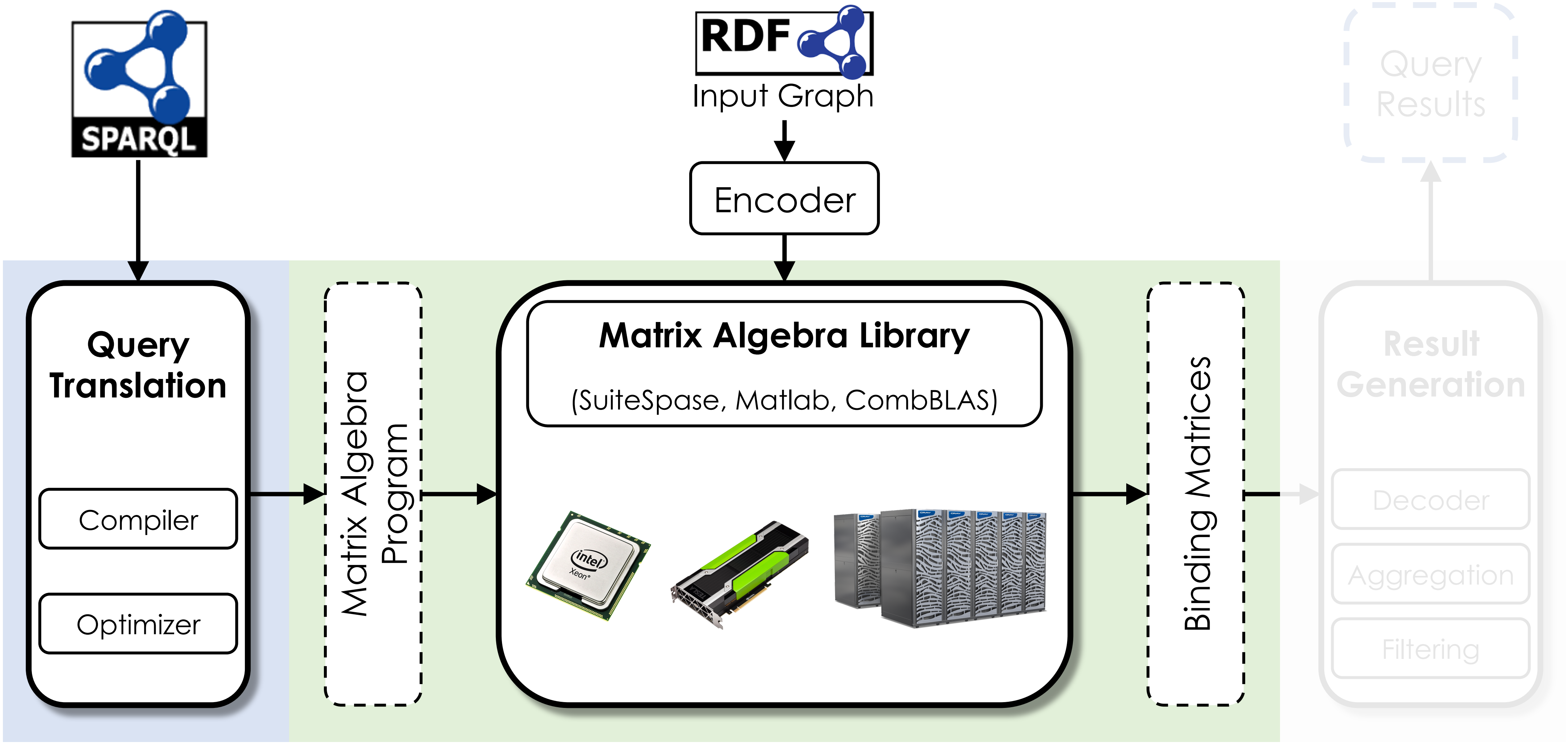
MAGiQ – Architecture



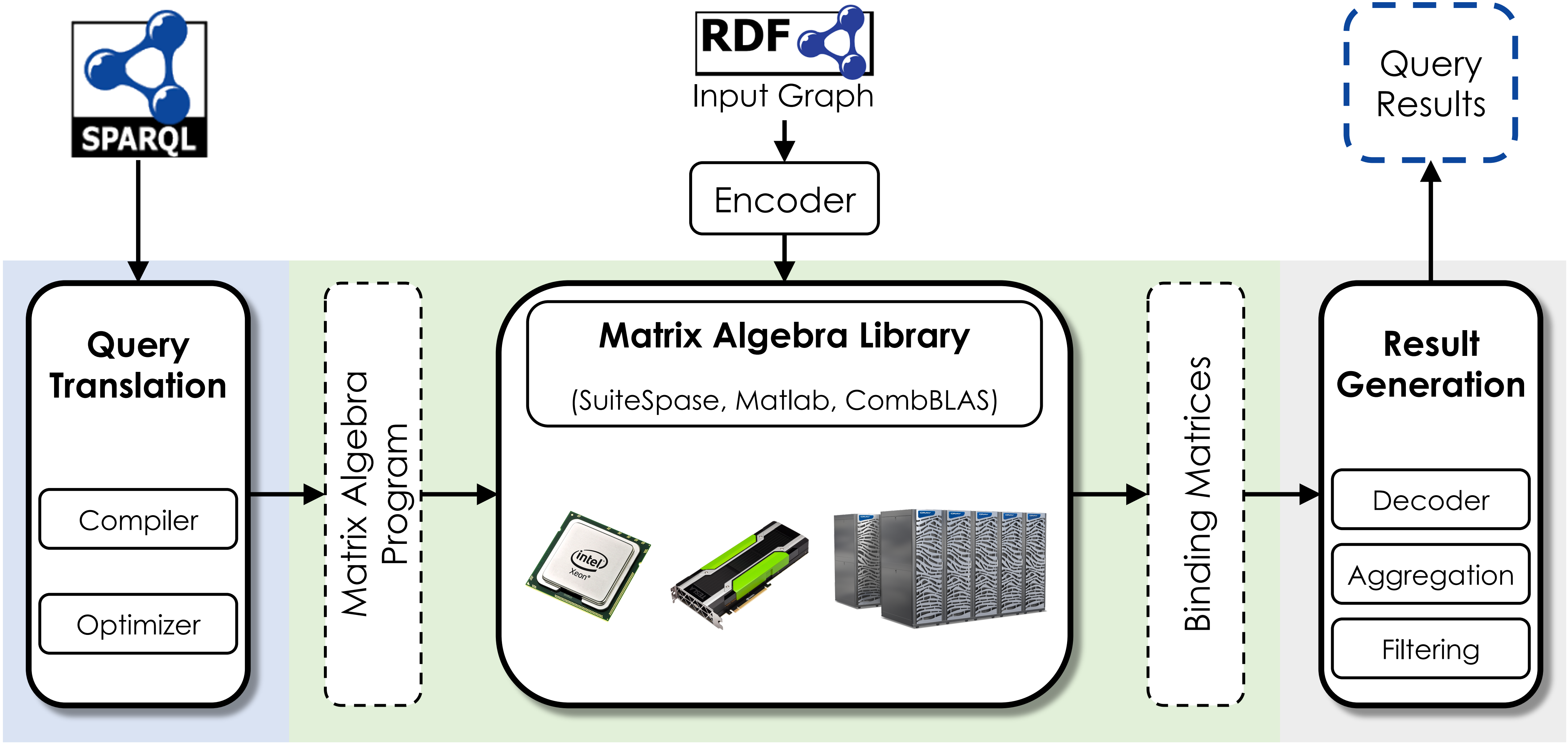
MAGiQ – Architecture



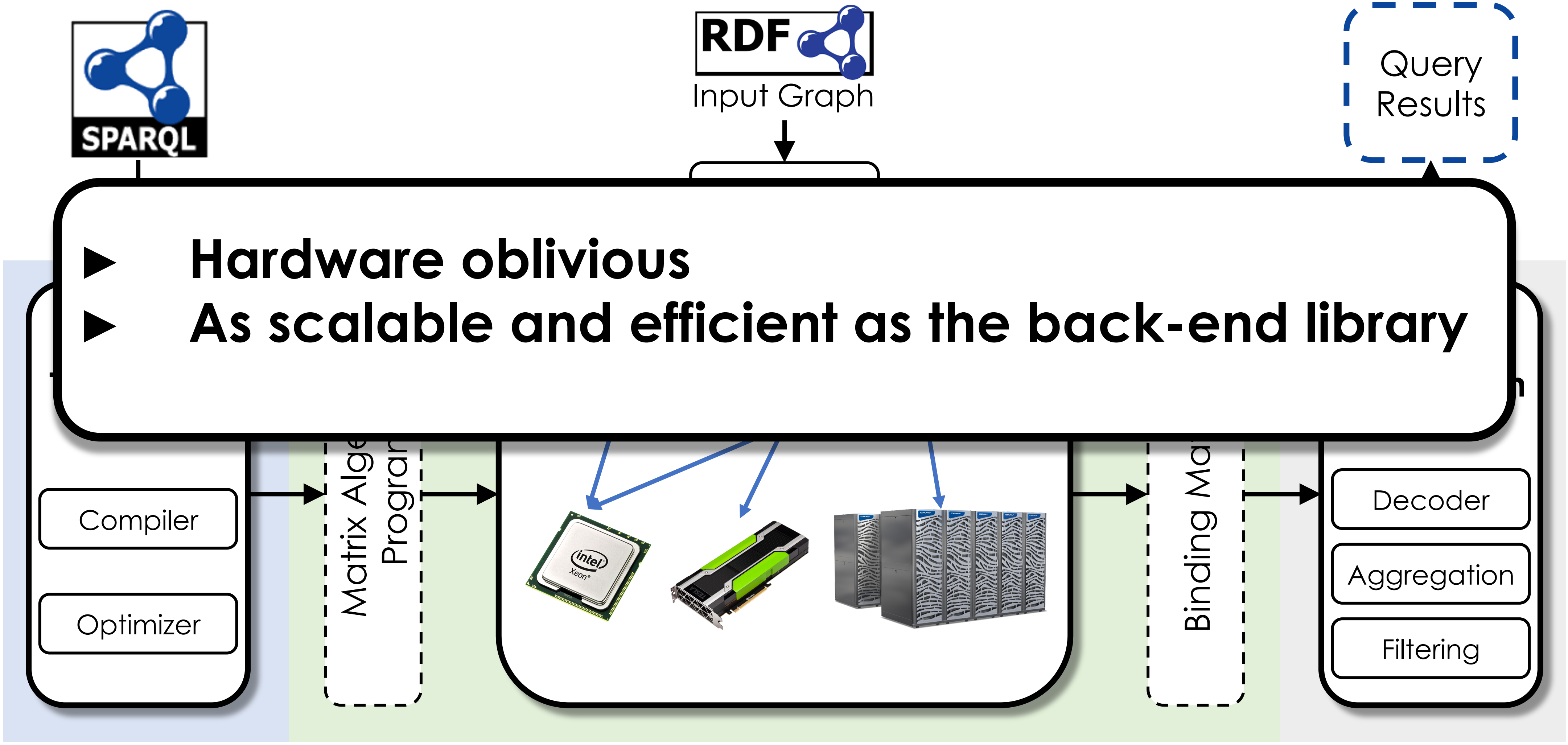
MAGiQ – Architecture



MAGiQ – Architecture



MAGiQ – Architecture



- ▶ **Hardware oblivious**
- ▶ **As scalable and efficient as the back-end library**

MAGiQ – Evaluation Setup

- Single machine: 2 x 14-core Intel Xeon E5-2680 @ 2.4Gh
  512GB
  NVIDIA Quadro P6000 GPU [Pascal, 24GB GDDR5X]

MAGiQ – Evaluation Setup

- Single machine: 2 x 14-core Intel Xeon E5-2680 @ 2.4Gh
  512GB
  NVIDIA Quadro P6000 GPU [Pascal, 24GB GDDR5X]

- Distributed-memory: Cray XC40

[6,174 Compute Nodes: 12,348 CPUs]
2 x 16-core Intel Xeon E5-2698 @ 2.3 GHz
128GB per Compute Node

MAGiQ – Datasets

Dataset	#Triples (M)	#Nodes (M)	#P
WatDiv-100M	109.23	10.28	85
YAGO2	284.30	60.70	98
WatDiv-1B	1,092.16	97.39	86
LUBM-1B	1,366.71	336,51	18
Bio2RDF	4,287.59	1,135.93	1,714
LUBM-10B	10,677.83	2,628.99	18
LUBM-512B	512,527.41	126,188.23	18

MAGiQ – Competing Engines

- Research:
  - **RDF-3X** [VLDB'08] – Relational, single machine, serial
  - **gStore** [VLDB'11] – Graph-based, single machine, serial
  - **AdPart** [VLDBJ'16]– Relational, distributed
  - **Wukong**[OSDI'16] – Graph-based, distributed, multithreaded
- Commercial:
  - **UrikaGD** – Specialized hardware appliance
  - **Virtuoso** – Relational, single machine, multithreaded

MAGiQ – Data-intensive Queries*

LUBM-1B (1.3B edges)

	Loading time (minutes)	Data-intensive query time (seconds)				Query time GeoMean (seconds)
		L1	L2	L3	L7	
RDF-3X [VLDB'08]	447	901	116	898	426	308
Wukong [OSDI'16]	57	11	10	11	42	15
MAGiQ(SuiteSparse)	16	173	38	108	155	102
MAGiQ(Matlab-CPU)	16	25	14	6	38	17
MAGiQ(Matlab-GPU)	16	**3**	**2**	**2**	**5**	**3**

*MAGiQ is slower for selective queries

MAGiQ – Data-intensive Queries*

LUBM-1B (1.3B edges)

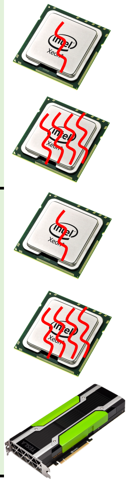
		Loading time (minutes)	Data-intensive query time (seconds)				Query time GeoMean (seconds)
			L1	L2	L3	L7	
RDF-3X	[VLDB'08]	447	901	116	898	426	308
Wukong	[OSDI'16]	57	11	10	11	42	15
MAGiQ(SuiteSparse)		16	173	38	108	155	102
MAGiQ(Matlab-CPU)		16	25	14	6	38	17
MAGiQ(Matlab-GPU)		16	**3**	**2**	**2**	**5**	**3**

*MAGiQ is slower for selective queries

MAGiQ – Data-intensive Queries*

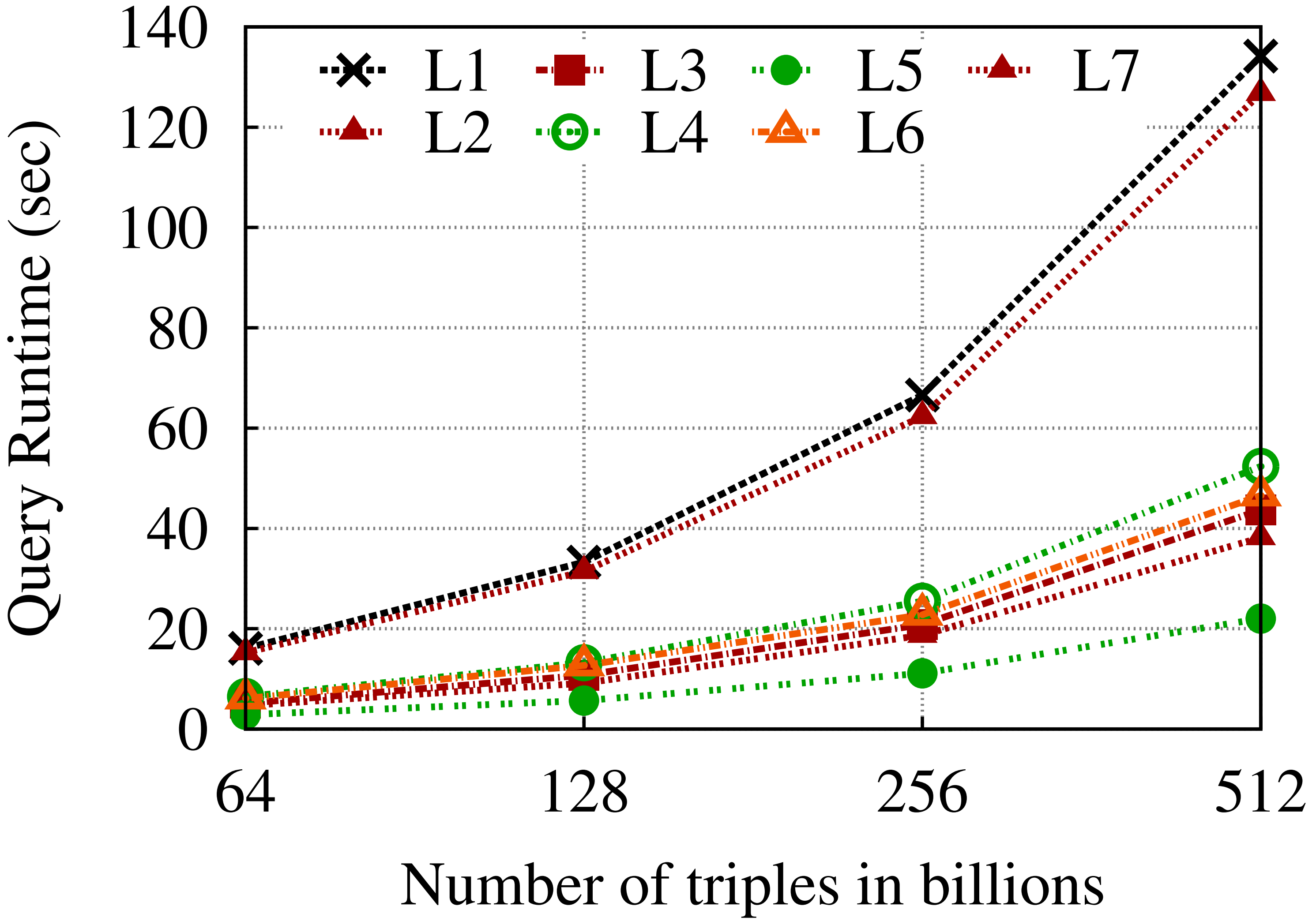
LUBM-1B (1.3B edges)

		Loading time (minutes)	Data-intensive query time (seconds) L1	L2	L3	L7	**Query time GeoMean** (seconds)
RDF-3X	[VLDB'08]	447	901	116	898	426	308
Wukong	[OSDI'16]	57	11	10	11	42	15
MAGiQ(SuiteSparse)		16	173	38	108	155	102
MAGiQ(Matlab-CPU)		16	25	14	6	38	17
MAGiQ(Matlab-GPU)		16	**3**	**2**	**2**	**5**	**3**



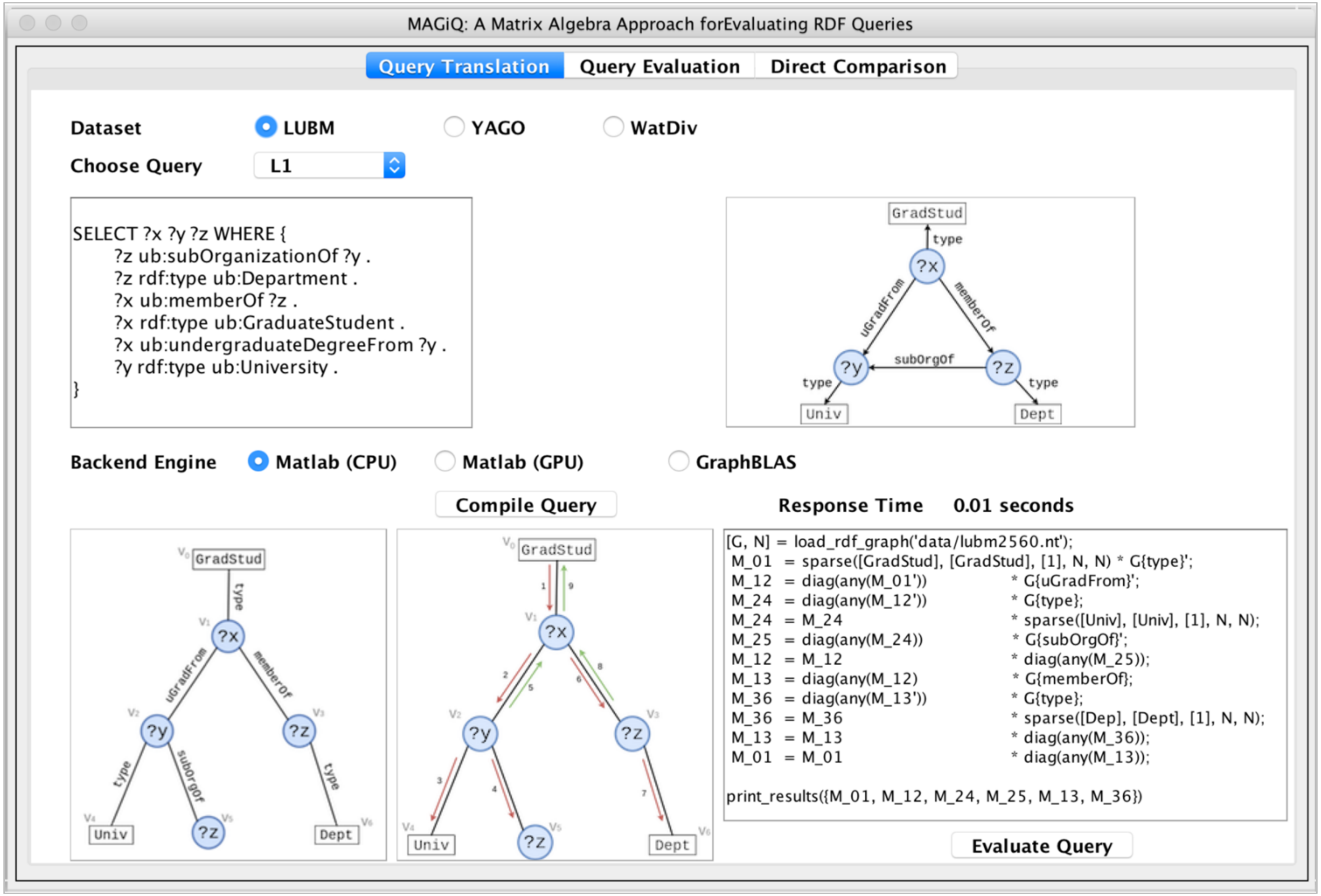
*MAGiQ is slower for selective queries

MAGiQ – Scalability on Cray XC40 Supercomputer

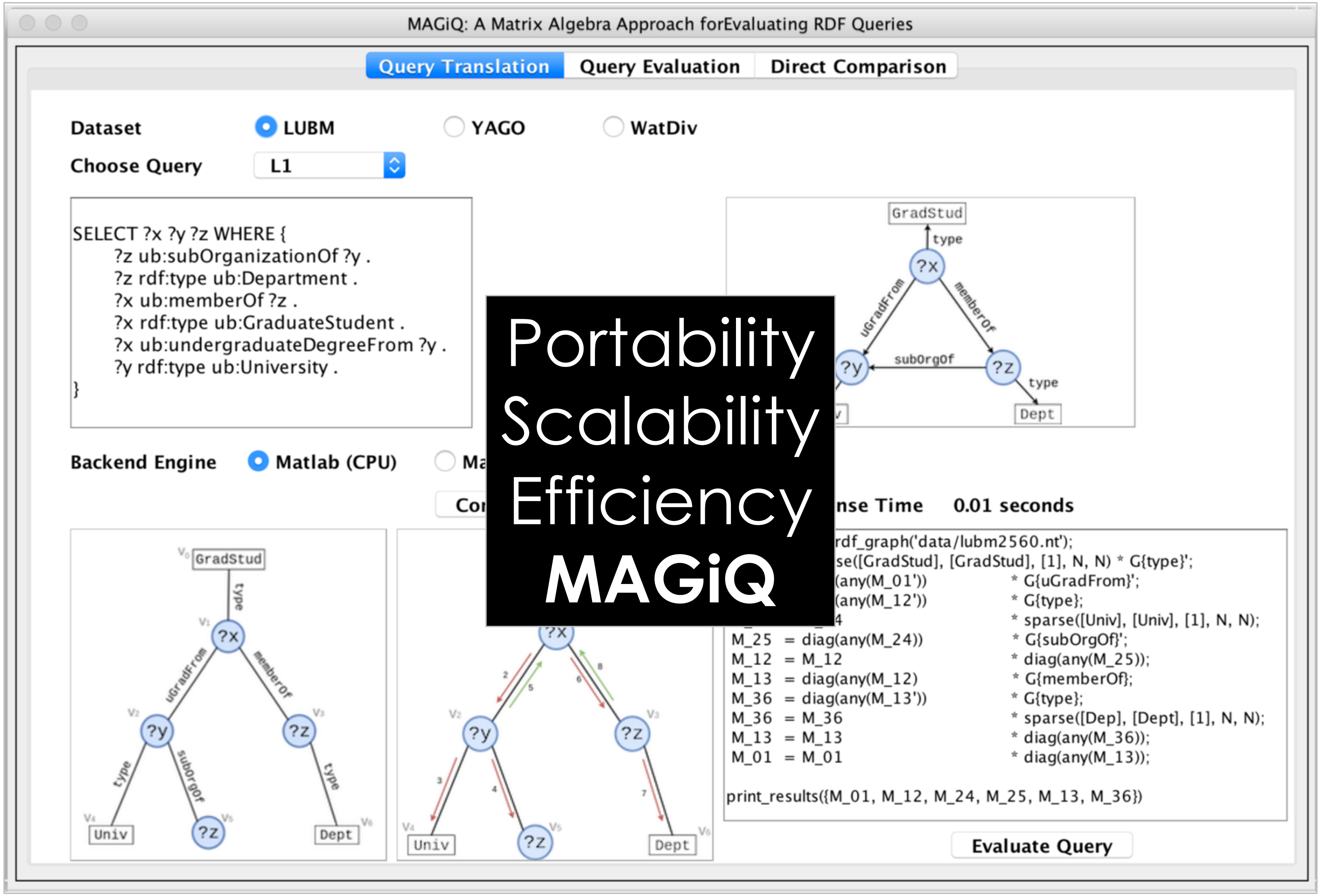


Data Scalability

2,048 compute nodes



MAGiQ: A Matrix Algebra Approach forEvaluating RDF Queries
Query Translation
Query Evaluation
Direct Comparison
Dataset
LUBM
YAGO
WatDiv
Choose Query
L1
SELECT ?x ?y ?z WHERE {
?z ub:subOrganizationOf ?y .
?z rdf:type ub:Department .
?x ub:memberOf ?z .
?x rdf:type ub:GraduateStudent .
?x ub:undergraduateDegreeFrom ?y .
?y rdf:type ub:University .
}
GradStud
type
?x
uGradFrom
memberOf
?y
subOrgOf
?z
type
type
Univ
Dept
Backend Engine
Matlab (CPU)
Matlab (GPU)
GraphBLAS
Compile Query
Response Time
0.01 seconds
[G, N] = load_rdf_graph('data/lubm2560.nt');
M_01 = sparse([GradStud], [GradStud], [1], N, N) * G{type}';
M_12 = diag(any(M_01')) * G{uGradFrom}';
M_24 = diag(any(M_12')) * G{type};
M_24 = M_24 * sparse([Univ], [Univ], [1], N, N);
M_25 = diag(any(M_24)) * G{subOrgOf}';
M_12 = M_12 * diag(any(M_25));
M_13 = diag(any(M_12)) * G{memberOf};
M_36 = diag(any(M_13')) * G{type};
M_36 = M_36 * sparse([Dep], [Dept], [1], N, N);
M_13 = M_13 * diag(any(M_36));
M_01 = M_01 * diag(any(M_13));
print_results({M_01, M_12, M_24, M_25, M_13, M_36})
Evaluate Query



Portability

Scalability

Efficiency

MAGiQ

Matrix algebra for RDF: it's MAGIQ

Panos.Kalnis @ KAUST.edu.sa

Extra slides

Extra:

- Main limitations:
  - No support for queries with variable predicates.
  - No support property path queries.
  - Slow performance for selective queries compared to index-based engines.

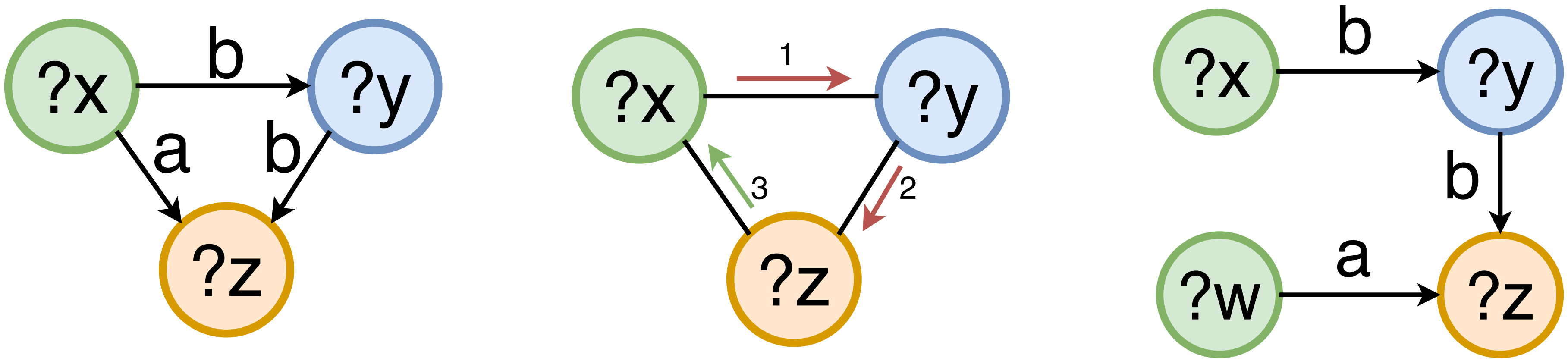
Extra: Slow for selective queries

LUBM-1B [data-intensive] query times (seconds)

	L1	**L2**	**L3**	**L7**	**GeoMean**
RDF-3X	901.4	115.6	898.2	426.2	307.6
Virtuoso	25.0	1,268.2	11.7	308.1	103.3
UrikaGD	5.8	2.4	1.9	7.0	3.7
Wukong	11.1	10.3	10.5	42.0	15.0
MAGiQ (SuiteSparse)	173.2	38.0	107.7	155.0	102.4
MAGiQ (Matlab-CPU)	24.9	14.3	6.1	38.2	17.0
MAGiQ (Matlab-GPU)	**3.2**	**2.1**	**1.5**	**5.4**	**2.7**

Extra: Cycles

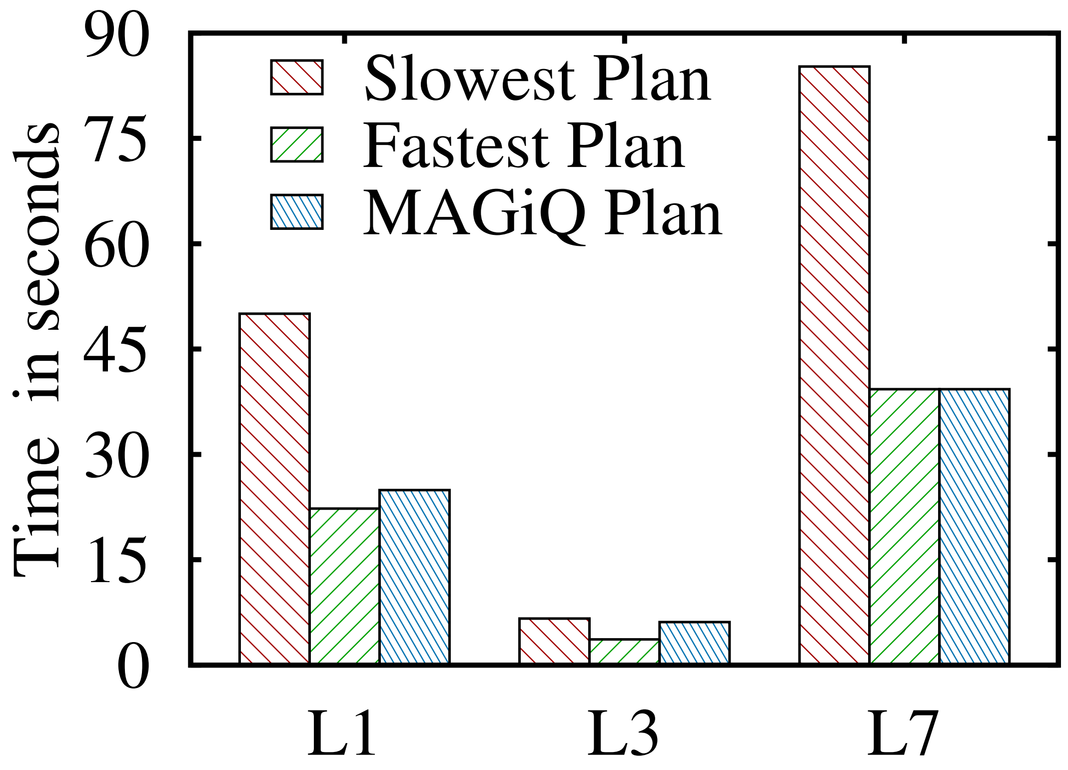
Cyclic queries



(a) (b) (c)

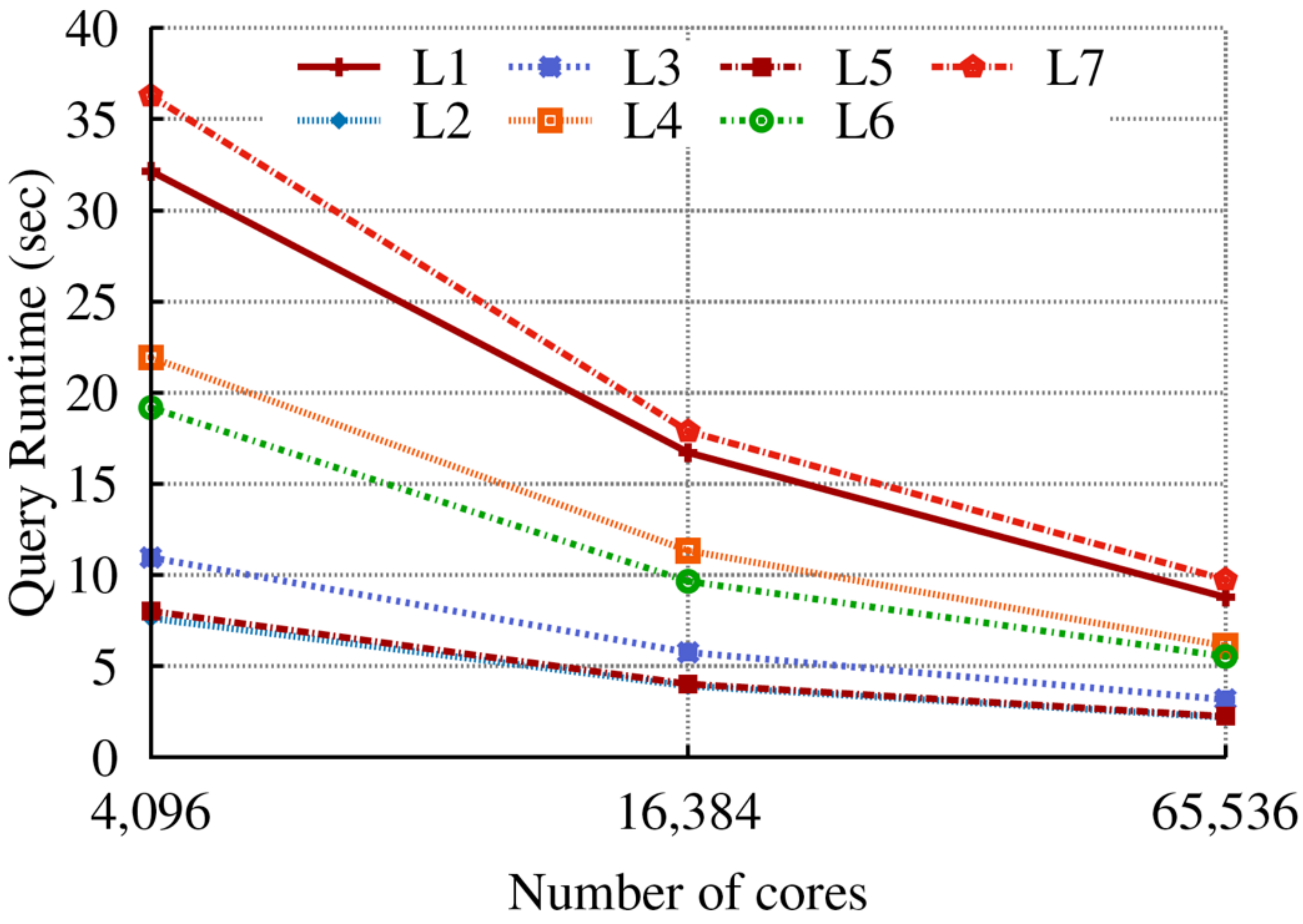
Extra:

Query planning



(a) MAGiQ (MATLAB-CPU)

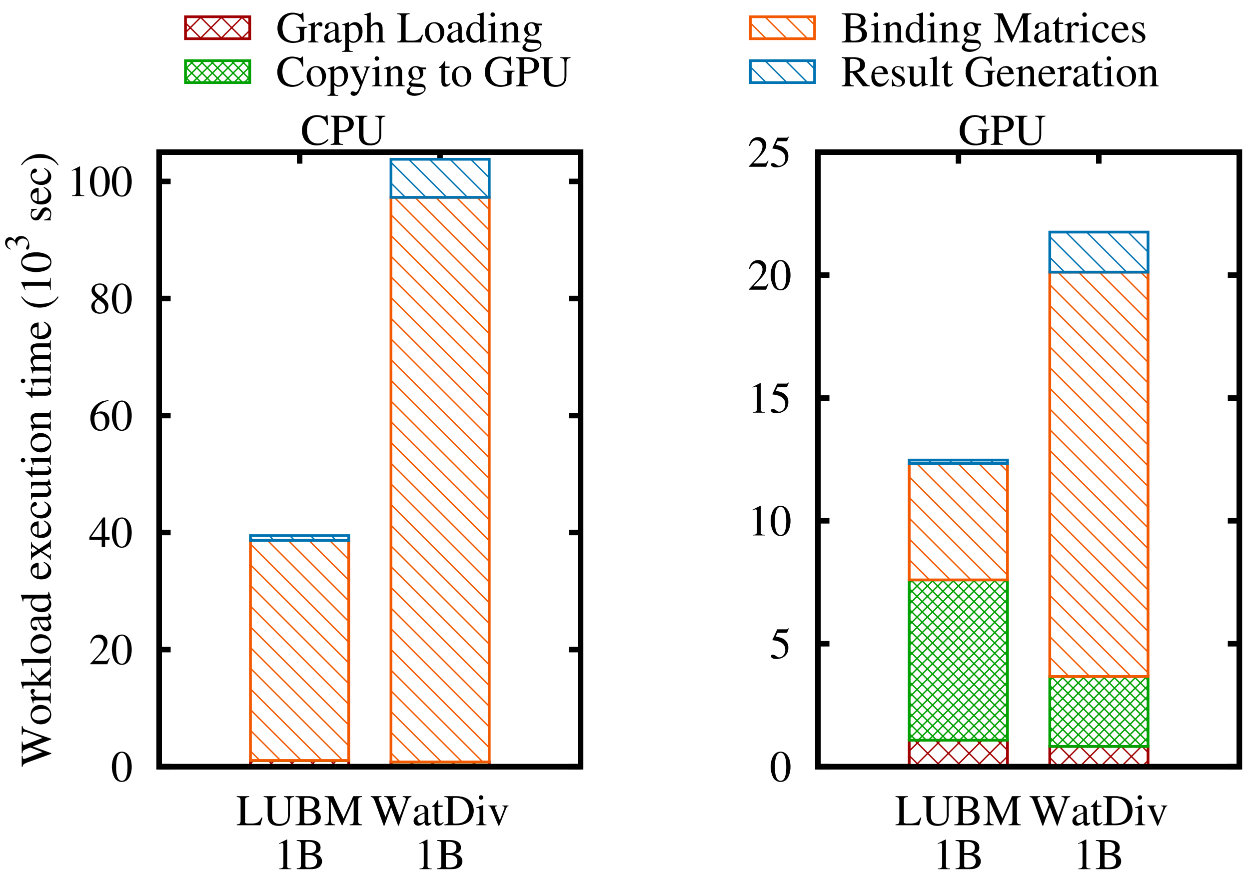
13B Triples - Shaheen



- CombBLAS
  - Scales with *sqrt*(p), where p: number of cores

Extra:

Workload evaluation



Extra:

LUBM-10B query times (seconds)

	L1	**L2**	**L3**	**L4**	**L5**	**L6**	**L7**
AdPart	5.12	**0.12**	**0.24**	**0.07**	**0.08**	3.51	4.84
MAGiQ (CombBLAS)	**3.08**	0.93	0.67	1.66	0.61	**1.36**	**3.79**

Extra:

Loading time (minutes)

Dataset	RDF-3X	gStore	Virtuoso	Wukong	MAGiQ
WatDiv-100M	18	40	9	4	**1**
YAGO2	78	63	50	9	**3**
LUBM-1B	447	n/a	191	57	**16**
Bio2RDF	n/a	n/a	331	n/a	**92**