Subgraph Search Over Massive Disk Resident Graphs

Peng Peng[1], Lei Zou[1], Lei Chen[2], Xuemin Lin[3], and Dongyan Zhao[1,4]

[1]Peking University,

[2]Hong Kong University of Science and Technology,

[3]University of New South Wales,

[4]Key Laboratory of Computational Linguistics (PKU)

Outline

- Background
- Index Technique
- Query Algorithm
- Experiments
- Conclusions & Future Work

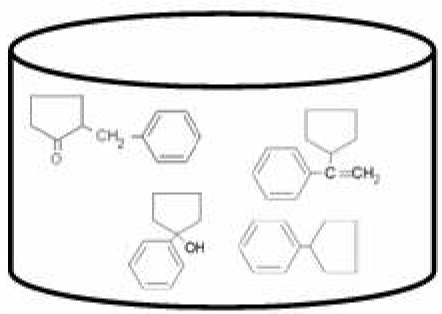
Outline

- Background
- Index Technique
- Query Algorithm
- Experiments
- Conclusions & Future Work

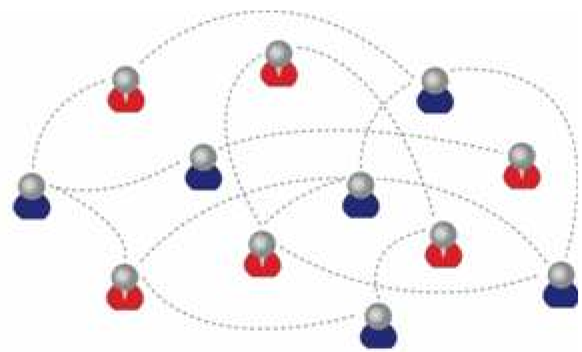
北京大學

Graph Database

As an alterative to relational database, graph database utilizes graph as the underlying model, which represents and stores information by nodes and connecting edges.

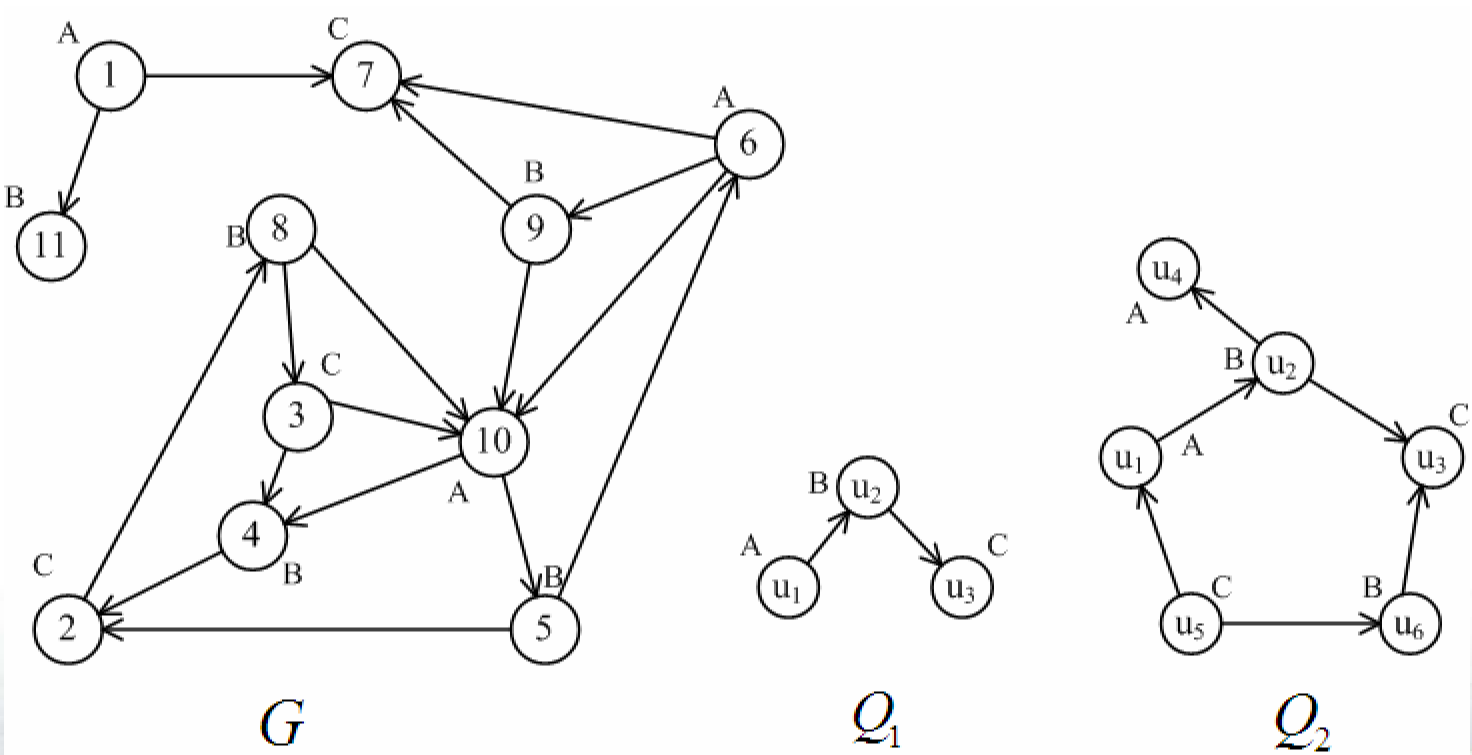


(a) A Large Number of Small-size Graphs



(b) A Single Large Graph

- In this paper, we focus on finding all embeddings of Q over a single large graph



Outline

- Background
- Index Technique
- Query Algorithm
- Experiments
- Conclusions & Future Work

Label of head	Label of tail	ID of head	ID of tail
A	C	1	7
A	B	1	11
C	B	2	8
C	B	3	4
C	A	3	10
B	C	4	2
B	C	5	2
B	A	5	6
A	C	6	7
A	B	6	9
A	A	6	10
B	C	8	3
B	A	8	10
B	C	9	7
B	A	9	10
A	B	10	4
A	B	10	5

A	A	6	10

A	B	1	11
A	B	6	9
A	B	10	4
A	B	10	5

A	C	1	7
A	C	6	7

B	A	5	6
B	A	8	10
B	A	9	10

B	C	4	2
B	C	5	2
B	C	8	3
B	C	9	7

C	A	3	10

C	B	2	8
C	B	3	4

- We divide all edges according to their labels.
- All of the edges with the same label information are organized together.

Index Structure

- For edges with the same labels, firstly we build two clustered B+-trees over it to save I/O cost.

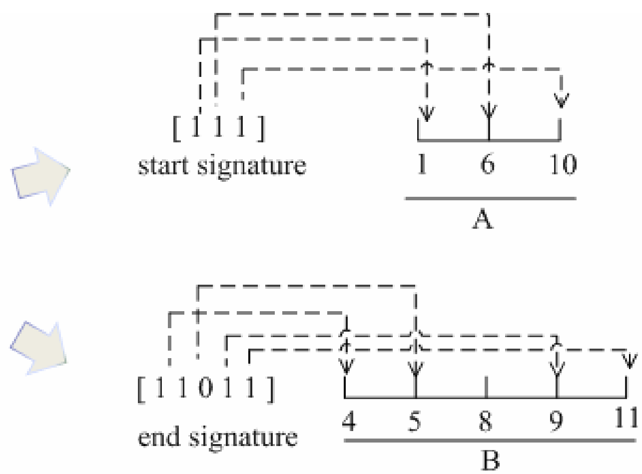
A	B	1	11
A	B	6	9
A	B	10	4
A	B	10	5

A	B	1	11
A	B	6	9
A	B	10	4
A	B	10	5

A	B	10	4
A	B	10	5
A	B	6	9
A	B	1	11

- Secondly, we propose two bitmap indexing structures for these edges.

A	B	1	11
A	B	6	9
A	B	10	4
A	B	10	5



- As discussed above, for the edges with the same labels information, we have assigned them four data structures, and they are shown as the following:

$\langle A, A\rangle$	[010]	[001]	$\overline{(6,10)}$	$\overline{(6,10)}$
$\langle A, B\rangle$	[111]	[11011]	$\overline{(1,11)}, \overline{(6,9)}, \overline{(10,4)}, \overline{(10,5)}$	$\overline{(10,4)}, \overline{(10,5)}, \overline{(6,9)}, \overline{(1,11)}$
$\langle A, C\rangle$	[110]	[001]	$\overline{(1,7)}, \overline{(6,7)}$	$\overline{(1,7)}, \overline{(6,7)}$

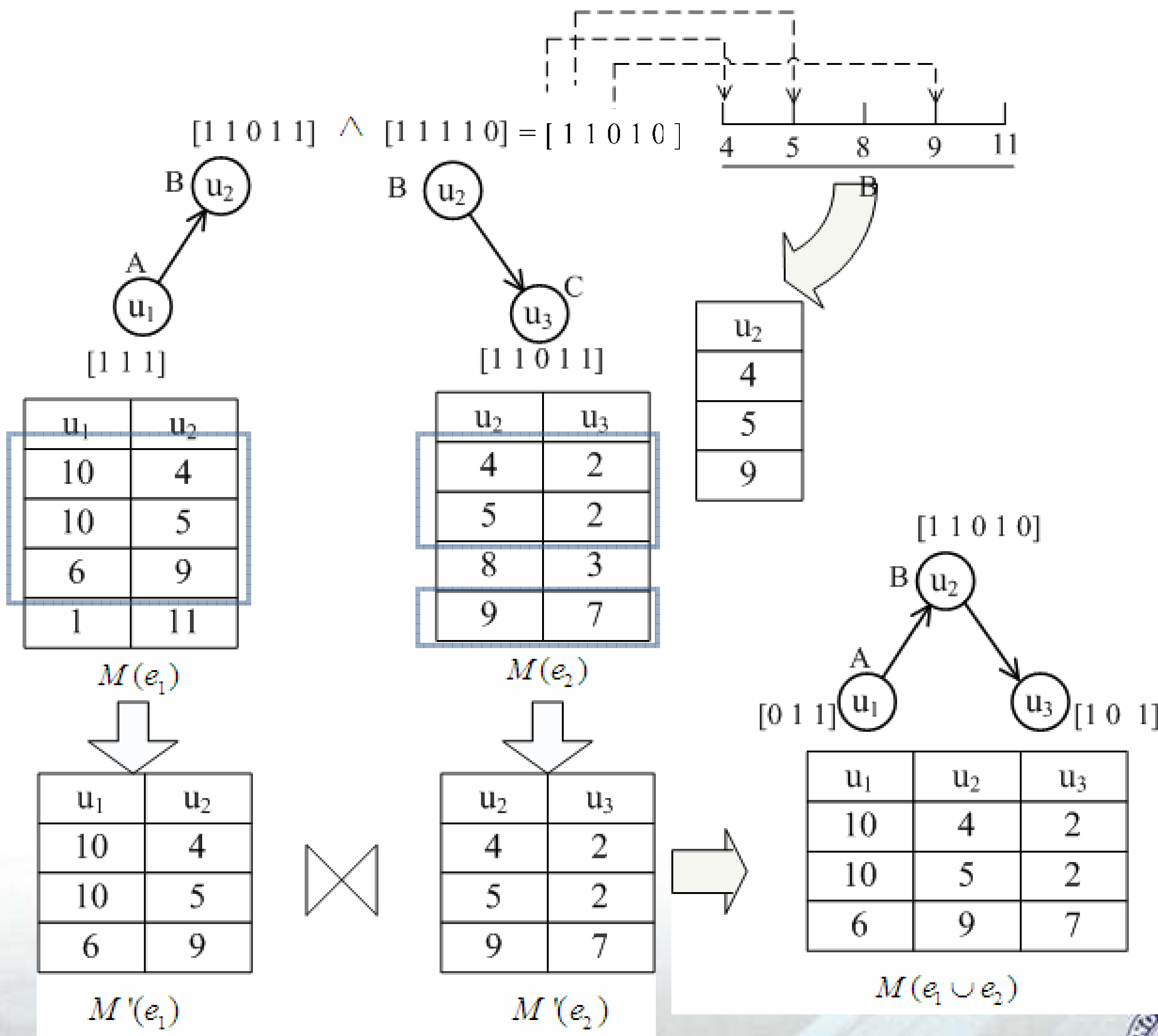
… …

Outline

- Background
- Index Technique
- Query Algorithm
- Experiments
- Conclusions & Future Work

Edge Join

- Let's consider the query graph Q_1 in our example, we propose Edge Join algorithm to handle it.


[1 1 0 1 1] ∧ [1 1 1 1 0] = [1 1 0 1 0]
4 5 8 9 11
B
A
u1
u2
[1 1 1]
C
u3
[1 1 0 1 1]
u1 u2
10 4
10 5
6 9
1 11
M(e1)
u2 u3
4 2
5 2
8 3
9 7
M(e2)
u2
4
5
9
[1 1 0 1 0]
[0 1 1]
[1 0 1]
u1 u2
10 4
10 5
6 9
M'(e1)
u2 u3
4 2
5 2
9 7
M'(e2)
u1 u2 u3
10 4 2
10 5 2
6 9 7
M(e1 ∪ e2)

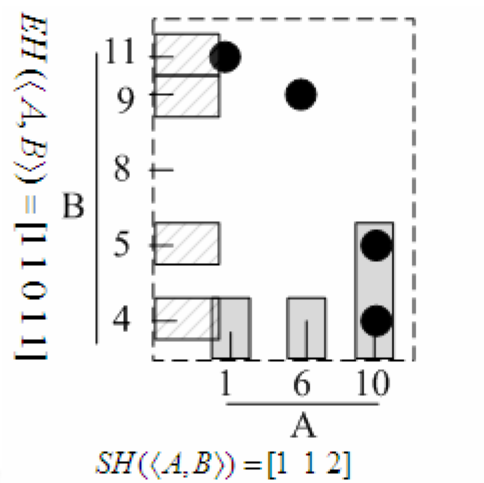
北京大学

AEP-based Query Algorithm

- For a complex query, we first find a set of subqueries above to cover Q. We call the subquery is ***a***djacent ***e***dge ***p***air (AEP for short) query, and propose an greedy strategy to find the cover of query Q.

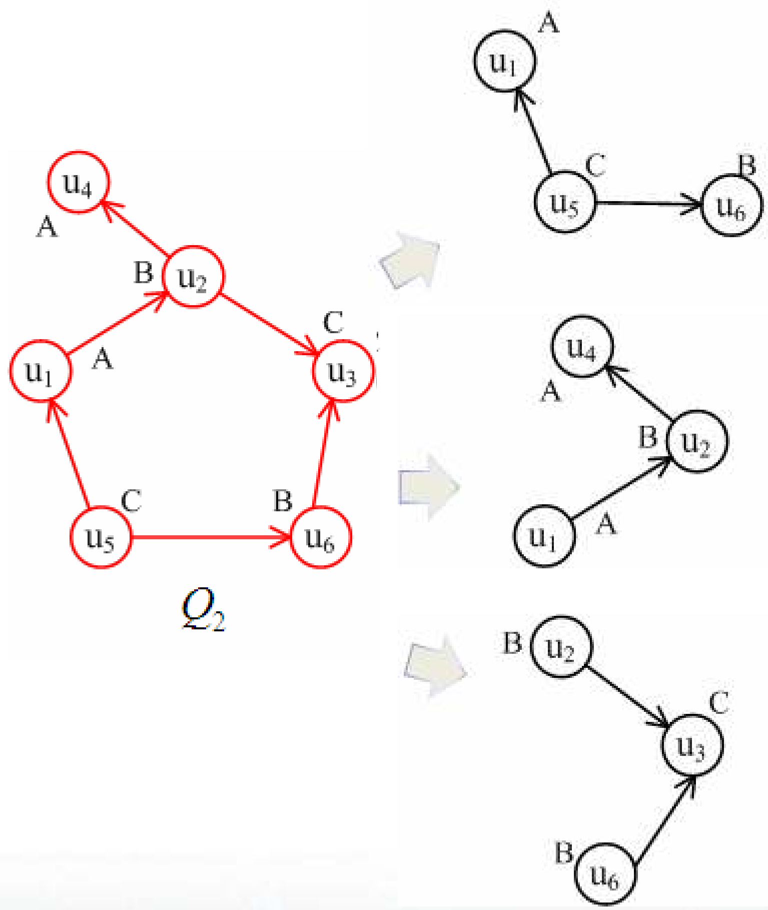
Cost Estimation

- For finding a better cover of the query, we propose two histograms to estimate the join cost.



(a) Histogram For Label Pair $\langle A, B\rangle$

AEP-based Query Algorithm



u5	u1
3	10

$\bowtie$

u5	u6
3	4

=

u1	u5	u6
10	3	4

u1	u2
6	9
10	5

$\bowtie$

u2	u4
5	6
9	10

=

u1	u2	u4
6	9	10
10	5	6

u2	u3
4	2
5	2

$\bowtie$

u6	u3
4	2
5	2

=

u2	u3	u6
4	5	2
5	4	2

u1	u2	u3	u4	u5	u6
10	5	2	6	3	4

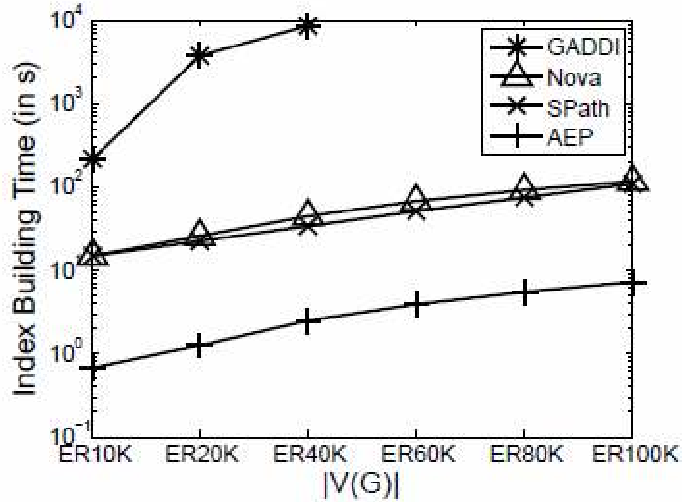
Outline

- Background
- Index Technique
- Query Algorithm
- Experiments
- Conclusions & Future Work

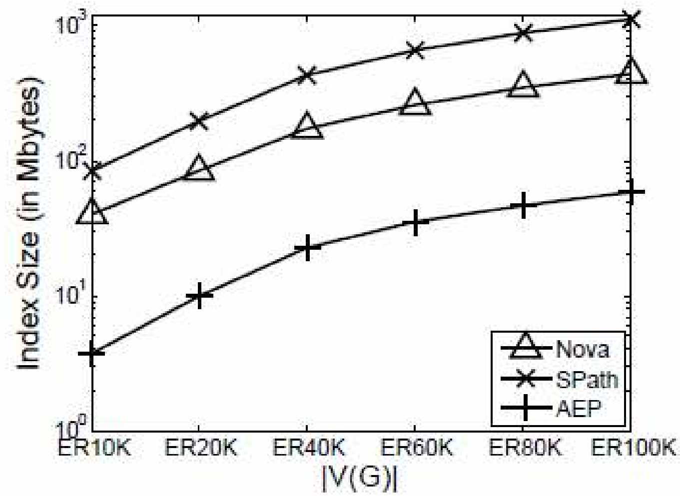
Datasets

	|V(G)|	|E(G)|	|L|
Erdos Renyi Model	10K~100K	50K~500K	250
Yago	368,587	543,815	45,450

Offline Performance



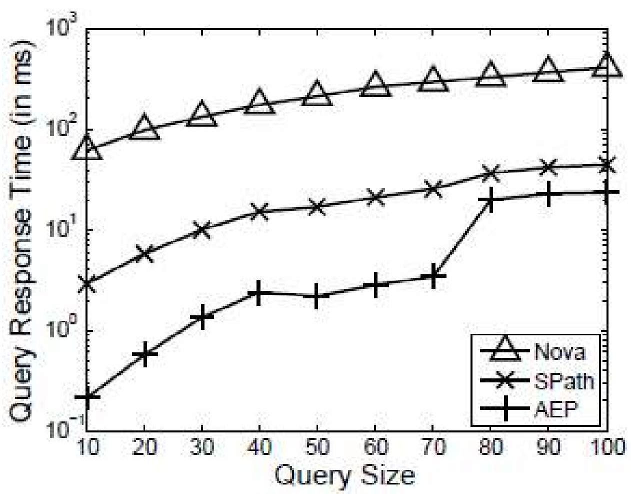
(a) Index Building Time (in seconds) over ER Graphs



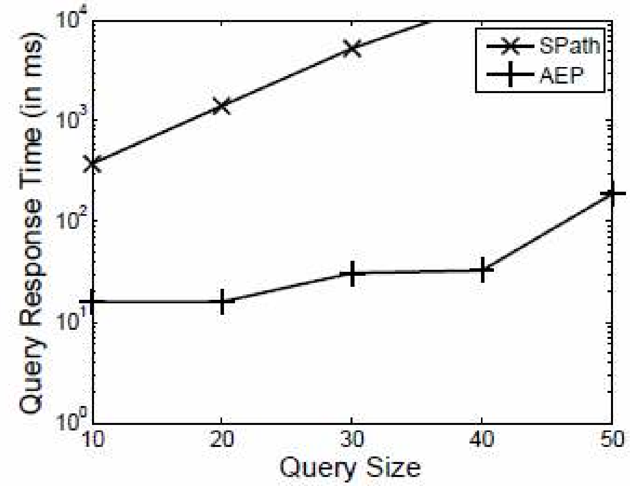
(b) Index Size (in Mbytes) over ER Graphs

北京大学

Online Performance



(a) Query Response Time (in milliseconds) over ER graphs



(b) Query Response Time (in milliseconds) over Yago graphs

Outline

- Background
- Overview of gStore
- Encoding Technique
- VS*-tree & Query Algorithm
- Experiments
- Conclusions & Future Work

Conclusions & Future Work

- In order to address subgraph query over a single large data graph G, in this paper, we
  - propose some novel index structure;
  - propose a subgraph query algorithm.

- In our experiment, we found that there might be too many intermediate results, so we will try to solve this problem in the future.

Q/A

Thank You!

pengpeng@icst.pku.edu.cn

北京大學