

Reachability Query Processing over Large Graphs

Jeffrey Xu Yu

Chinese University of Hong Kong, China
yu@se.cuhk.edu.hk

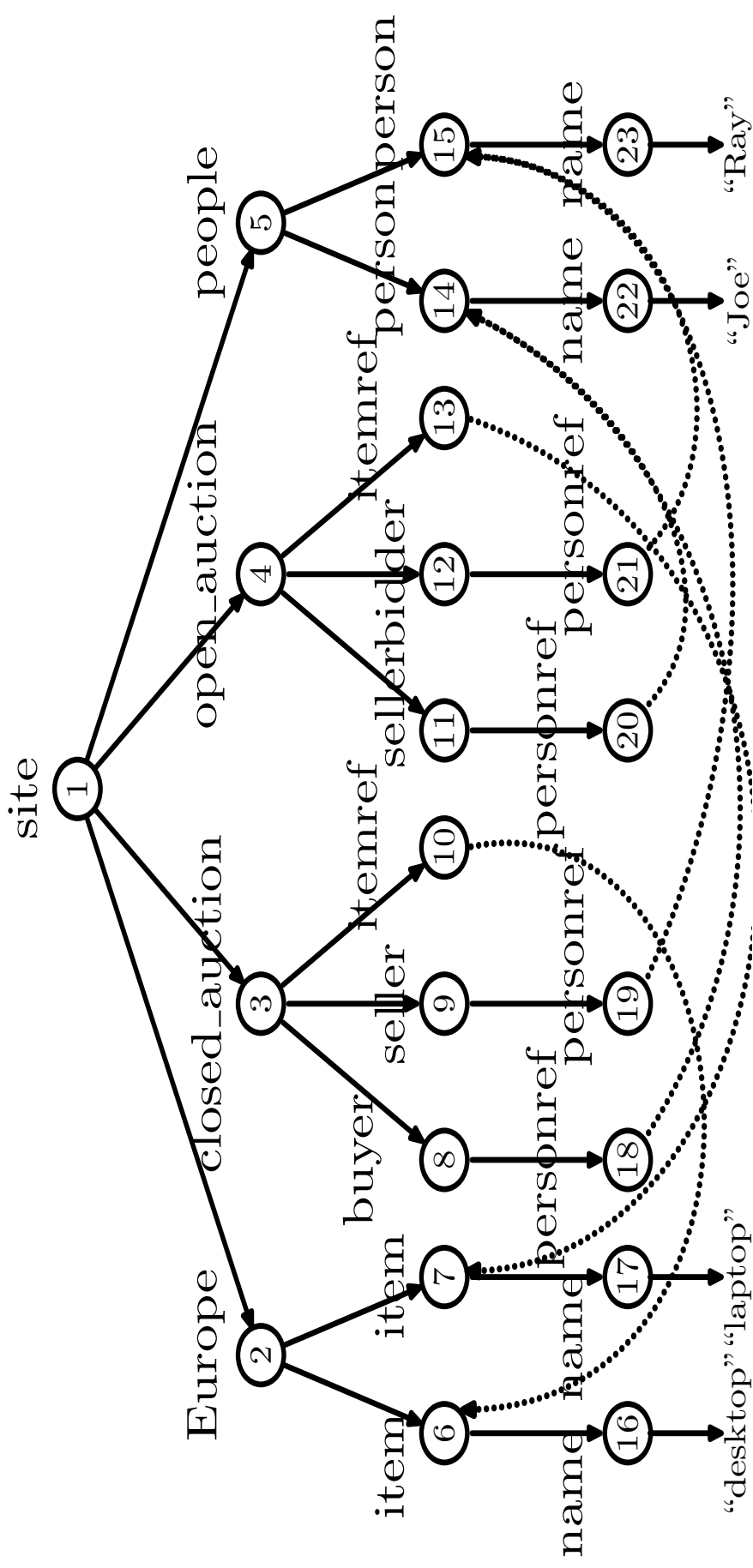
Overview

- Introduction of Reachability Queries
- An Existing Interval Based Approach
- Our 2-Hop Based Approach

Reachability Query

- Graph has great expressive power to describe the complex relationships among data objects.
- There is a huge amount of data on Web.
- XML allows users to model data as trees or to view data as graphs with two different links, the parent-child links and ID/IDREF. There are XLink and XPointer.
- Given two types of elements in a graph, A and D , a reachability query, denoted $A \rightsquigarrow D$, is to find all elements of the type D that are reachable from some elements in the type A .

An XML Example



- Find all seller's names ($seller \rightsquigarrow name$).

Support Reachability Queries Using XQuery

- Consider the query of finding all seller's names

(seller \rightsquigarrow name).

- An XQuery Solution

```
for $s in //seller  
let $n := for $p in //person  
           where $s/personref/@idref = $p/@id  
return $n/name
```

- It requires users to fully understand the schema.
- It may require many joins.

XML Structural Join

- Structural join techniques for processing $A//D$ cannot be directly applied to reachability queries, $A \rightsquigarrow D$.
 - $A//D$ does not traverse the cross-document links supported by ID/IDREF.
 - The existing algorithms for handling $A//D$ deals with the ancestor/descendants relationships in a tree structure.
 - $A//D$ is a special case of \rightsquigarrow when data is a tree.

Labeling-Based Approaches

- In order to efficiently process reachability queries over graph database, one strategy is to assign labels onto graph nodes.
- Given two nodes u and v ,
 $u \rightsquigarrow v$ if and only if $P(\text{code}(u), \text{code}(v)) = \text{true}$.
- For directed graphs, there are two labelings:
 - interval based labeling, and
 - 2-hop based labeling.

An Interval Based Approach

An Interval Based Labeling

- Agrawal, Borgida and Jagadish proposed an interval based labeling for **directed acyclic graphs** in SIGMOD'89.
- Wang et al. extended it to support **directed graphs** in APWeb'05.
 - Construct a directed acyclic graph G' by condensing a maximal strongly connected component in G as a node in G' .
 - Generate interval based labels for G' .
 - All nodes in a strongly connected component in G share the same label assigned to the corresponding representative node condensed in G' .

An Interval Based Algorithm (1)

- By Wang et al. (APWeb'05)
- Consider two lists, A and D .
- A encodes each node v as $(v, s : e)$ where $[s, e]$ is an interval for node v . **A node may have multiple intervals.** If a node of A has n intervals, it has n entries in A . A is sorted on $[s, e]$.
- D encodes each node v as (v, po_v) , where po_v is the postorder of v . D is sorted.
- Consider $seller \rightsquigarrow name (A \rightsquigarrow D)$. There are

$$A = \{(11, 2 : 3), (9, 6 : 9), (11, 14 : 15)\}$$

$$D = \{(22, 2), (23, 6), (16, 10), (18, 10)\}$$

An Interval Based Algorithm (2)

- With the two sorted lists, the Interval approach scans the two lists for the results of $A \rightsquigarrow D$.

$$A = \{ \underline{(11, 2 : 3)}, \underline{(9, 6 : 9)}, \underline{(11, 14 : 15)} \}$$

$$D = \{ \underline{(22, 2)}, \underline{(23, 6)}, \underline{(16, 10)}, \underline{(18, 10)} \}$$

- The processing cost depends on the sizes of the two lists.
- The total list size depends on the size of the interval based labeling.

A 2-Hop Based Approach

2-hop Labeling (1)

- 2-hop labeling was proposed by Cohen et al. in SODA'02.
- Intuitively, if $u \rightsquigarrow v$, there is w such as $u \rightsquigarrow w$ and $w \rightsquigarrow v$.
- The key issue is how to minimize the size of such labeling.
- Recall: a transitive closure of G can be used to answer if $u \rightsquigarrow v$, but the size of such transitive closure is large.

2-hop Labeling (2)

- Given a graph $G(V, E)$, assign every node $v \in V$ a label $L(v) = (L_{in}(v), L_{out}(v))$. $u \rightsquigarrow v \leftrightarrow L_{out}(u) \cap L_{in}(v) \neq \emptyset$.
- $11 \rightsquigarrow 22$, because $L_{out}(11) \cap L_{in}(22) = \{14\} \neq \emptyset$.

\mathcal{T}	v	L_{in}	L_{out}	\mathcal{T}	v	L_{in}	L_{out}
item	7	2, 4	\emptyset	person	15	3, 4, 15	15
buyer	8	3, 8	8	name	16	2, 3, 6	\emptyset
seller	9	3	15	name	17	2, 4, 7	\emptyset
seller	11	4	14	name	22	3, 4, 8, 14	\emptyset
bidder	12	4	15	name	23	3, 4, 15	\emptyset
item	6	2, 3, 6	6	person	14	3, 4, 8, 14	15

2-hop Labeling (3)

- Consider $seller \rightsquigarrow name$.
- We have types like *seller* and *name*.
- We have centers w , in $L_{in} \cup L_{out}$, which connect two points u and v as $u \rightsquigarrow w$ and $w \rightsquigarrow v$.

T	v	L_{in}	L_{out}	T	v	L_{in}	L_{out}
item	7	2, 4	\emptyset	person	15	3, 4, 15	15
buyer	8	3, 8	8	name	16	2, 3, 6	\emptyset
seller	9	3	15	name	17	2, 4, 7	\emptyset
seller	11	4	14	name	22	3, 4, 8, 14	\emptyset
bidder	12	4	15	name	23	3, 4, 15	\emptyset
item	6	2, 3, 6	6	person	14	3, 4, 8, 14	15

2-Hop Based Labeling

Cohen's Algorithm (1)

- The minimal 2-hop cover problem is NP-hard. The resulting size of Cohen's algorithm is larger than the optimal at most $O(\log n)$.
- Let T be an edge-transitive closure of a given graph G .
- Let $B_C(V_C, E_C)$ be a bipartite graph where $V_C = V_{C_{in}} \cup V_{C_{out}}$, for each node w (center) in the graph, such as

$$V_{C_{in}} = \{u \mid (u, w) \in T\} \cup \{w\} \quad (1)$$

$$V_{C_{out}} = \{v \mid (w, v) \in T\} \cup \{w\} \quad (2)$$

and

$$E_C \in V_{C_{in}} \times V_{C_{out}} \quad (3)$$

Cohen's Algorithm (2)

- $T' \leftarrow T$.
- Repeat the following until T' becomes empty.
 - Select a node w , regarding to its center graph $B_C(V_C, E_C)$, which has a subgraph where $V = V_{in} \cup V_{out}$, from B_C , with the max ratio of

$$\max_{\substack{V_{in} \subseteq V_{C_{in}} \\ V_{out} \subseteq V_{C_{out}} \\ E \subseteq E_C}} \frac{|E \cap T'|}{|V_{in}| + |V_{out}|} \quad (4)$$

where $E = V_{in} \times V_{out}$.

- Let $T' \leftarrow T' \setminus E$.

Our Algorithm

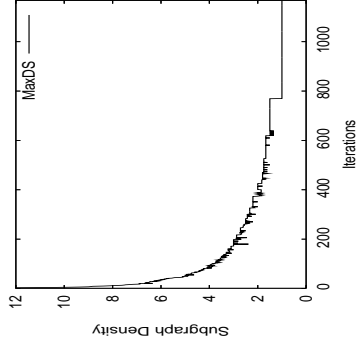
- $T' \leftarrow T$.
- Repeat the following until T' becomes empty.
 - Select a node w , regarding to its center graph $B_C(V_C, E_C)$, which has a subgraph where $V = V_{in} \cup V_{out}$, from B_C , with the max ratio of

$$\max_{\substack{V_{in} \subseteq V_C \\ V_{out} \subseteq V_C \\ E \subseteq E_C}} |E \cap T'| \quad (5)$$

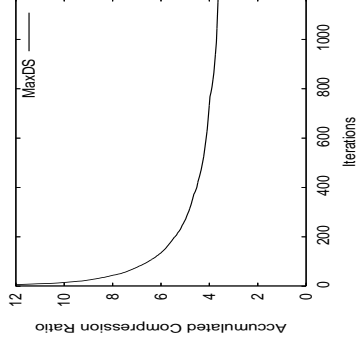
where $E = V_{in} \times V_{out}$.

- Let $T' \leftarrow T' \setminus E$.

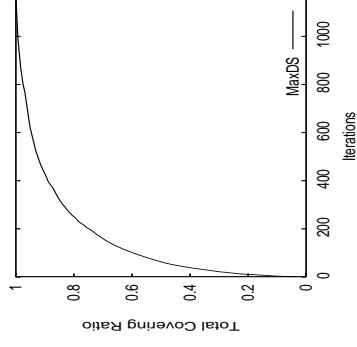
Differences Between Cohen's and Ours



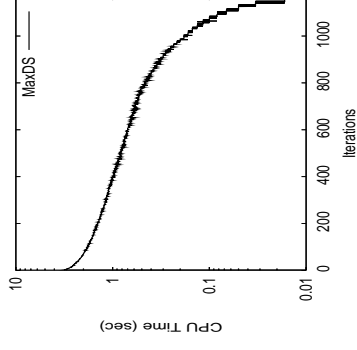
(a) Density



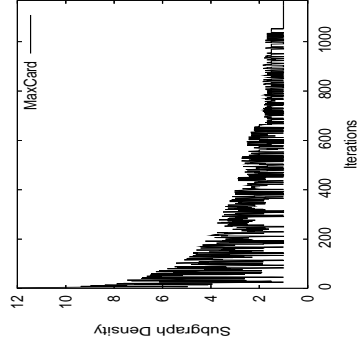
(b) Compression



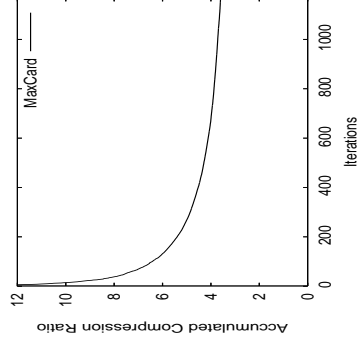
(c) Coverage



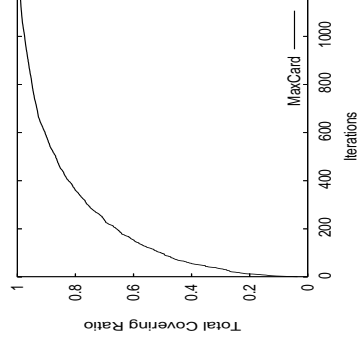
(d) CPU



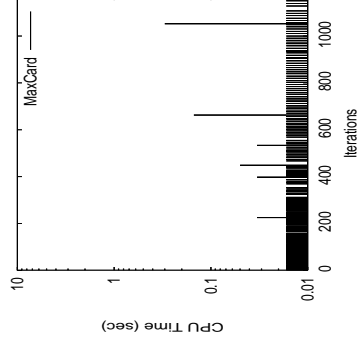
(e) Density



(f) Compression



(g) Coverage



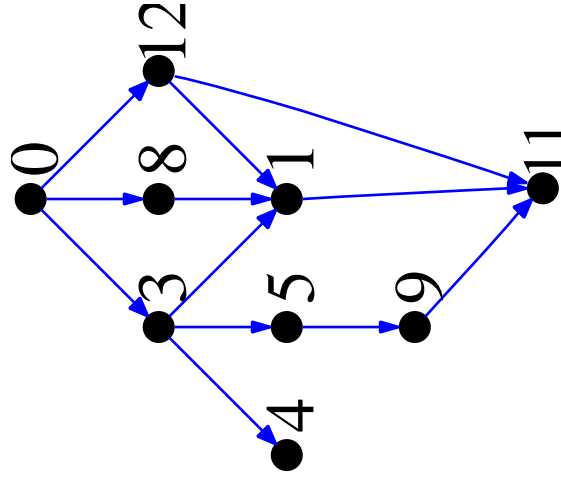
(h) CPU

- Tested with a DAG ($|V| = 2,000$, $|E| = 4,000$)

The Features of Our Algorithm

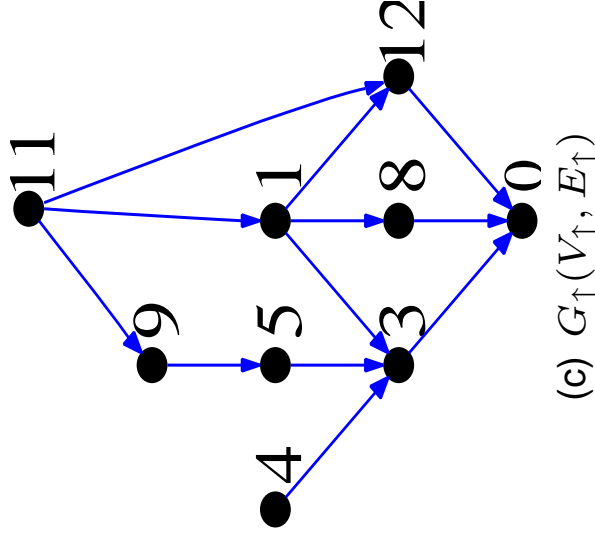
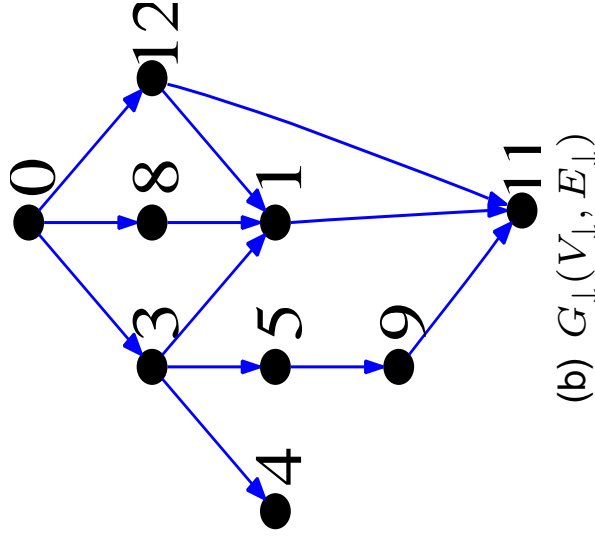
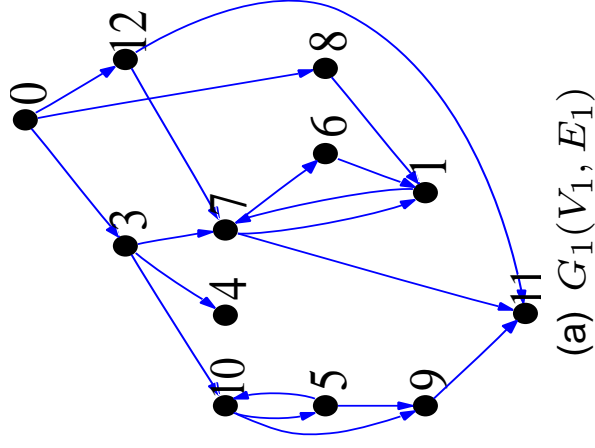
- Do not generate transitive closure.
- Map the 2-hop problem onto a two-dimensional grid, and
- Compute 2-hop labeling using operations against rectangles with help of a R-tree.

An Example



v	L_{in}	L_{out}	v	L_{in}	L_{out}
0	\emptyset	0, 3	8	0	1
1	1, 3	1	9	3, 9	9
3	3	3	11	1, 3, 9	\emptyset
4	3	\emptyset	12	0	1
5	3	9			

Utilizing Internal-Based Labeling



- A directed graph G_1 .
- Its two directed acyclic graphs, G_{\downarrow} and G_{\uparrow} .

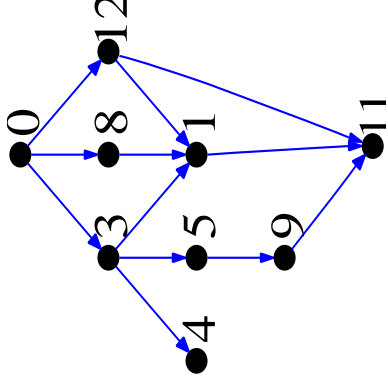
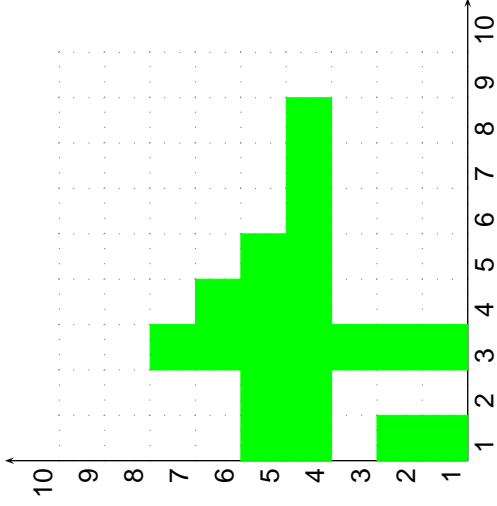
Interval Based Codes for G_{\downarrow} and G_{\uparrow}

w	G_{\downarrow}		G_{\uparrow}	
	$po_{\downarrow}(w)$	$I_{\downarrow}(w)$	$po_{\uparrow}(w)$	$I_{\uparrow}(w)$
0	9	[1,9]	4	[4,4]
1	1	[1,1],[3,3]	3	[1,5]
3	6	[1,6]	5	[4,5]
4	2	[2,2]	9	[4,5],[9,9]
5	5	[3,5]	6	[4,6]
8	7	[1,1],[3,3],[7,7]	1	[1,1],[4,4]
9	4	[3,4]	7	[4,7]
11	3	[3,3]	8	[1,8]
12	8	[1,1],[3,3],[8,8]	2	[2,2],[4,4]

Reachability Map

- Given a graph G_{\downarrow} .
- Consider a Reachability Map, M , size of $|V(G_{\downarrow})| \times |V(G_{\downarrow})|$.
- Given two nodes, u and v , in G_{\downarrow} .
- A function $f(u, v)$ maps the node pair (u, v) onto a grid $(x(v), y(u))$ in M , where $x(v) = op_{\downarrow}(v)$ and $y(u) = op_{\uparrow}(u)$.
- The grid value of $f(u, v)$ is 1, if $u \rightsquigarrow v$, otherwise 0.

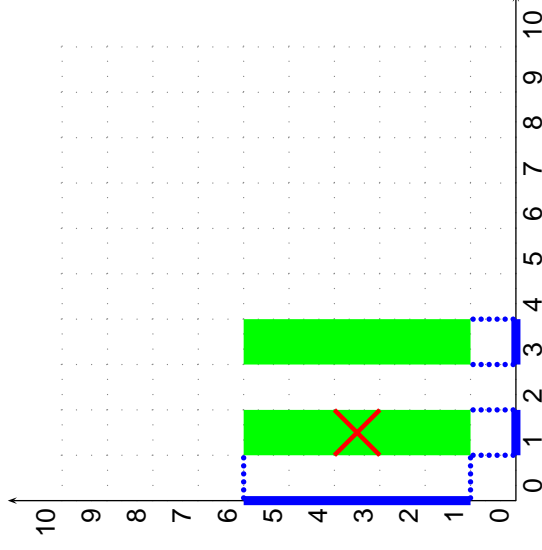
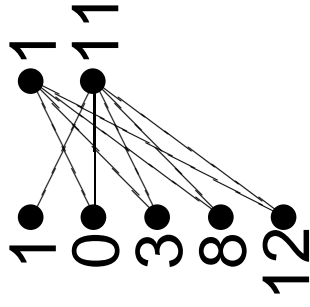
A Reachability Map Example



p	$f(p)$	p	$f(p)$	p	$f(p)$
$0 \rightsquigarrow 1$	(1, 4)	$0 \rightsquigarrow 3$	(6, 4)	$0 \rightsquigarrow 4$	(2, 4)
$0 \rightsquigarrow 5$	(5, 4)	$0 \rightsquigarrow 8$	(7, 4)	$0 \rightsquigarrow 9$	(4, 4)
$0 \rightsquigarrow 11$	(3, 4)	$0 \rightsquigarrow 12$	(8, 4)	$1 \rightsquigarrow 11$	(3, 3)
$3 \rightsquigarrow 1$	(1, 5)	$3 \rightsquigarrow 4$	(2, 5)	$3 \rightsquigarrow 5$	(5, 5)
$3 \rightsquigarrow 9$	(4, 5)	$3 \rightsquigarrow 11$	(3, 5)	$5 \rightsquigarrow 9$	(4, 6)
$5 \rightsquigarrow 11$	(3, 6)	$8 \rightsquigarrow 1$	(1, 1)	$8 \rightsquigarrow 11$	(3, 1)
$9 \rightsquigarrow 11$	(3, 7)	$12 \rightsquigarrow 1$	(1, 2)	$12 \rightsquigarrow 11$	(3, 2)

Bipartite Graph and Its Rectangles

- Let $B_C(V_C, E_C)$ be a bipartite graph for a center node w in G_{\downarrow} .
- Let M be the reachability map.
- A $Rect(w)$ function maps B_C onto M .
- Consider an example for $w = 1$.



Revisit Our Algorithm

- Let Δ be the covered area in M .
- Iteratively select a node w which has largest area of $Rect(w) - \Delta$.
- Repeat the above until $\Delta = \bigcup_{w \in V(G_{\downarrow})} Rect(w)$.

Reachability Query Processing Based on 2-Hop Based Labeling

A Join-Index Based on 2-Hop Labeling

- We propose a join-index to process reachability queries.
- The join-index consists of a table and a B+-tree.

A Center Table

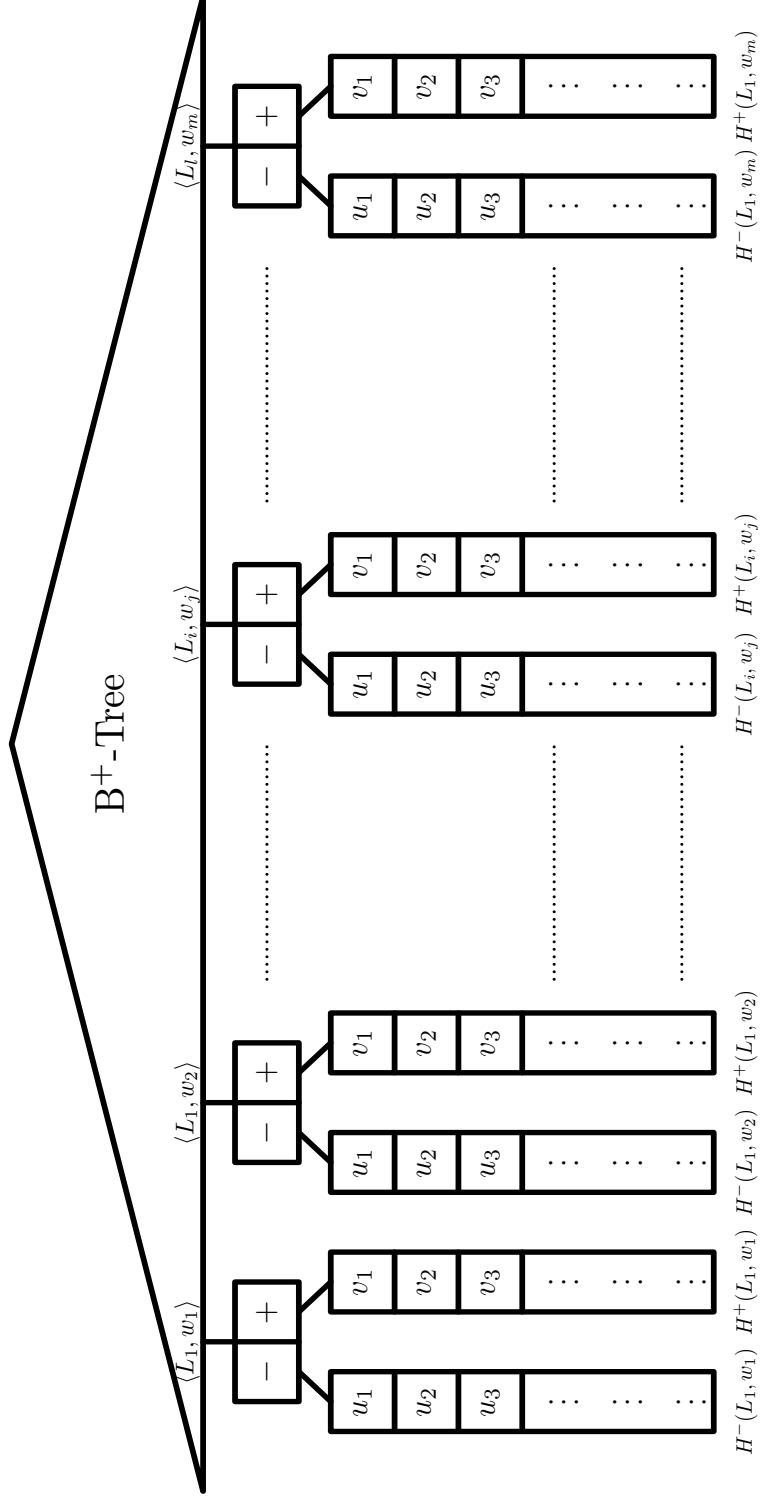
- A center table is built for, $W(A, D)$, every two types, A and D .

$$W(A, D) = \left(\bigcup_{\lambda(a)=A} L_{out}(a) \right) \cap \left(\bigcup_{\lambda(d)=D} L_{in}(d) \right)$$

- Here, $W(A, D)$ represents the set of centers, w , for $A \rightsquigarrow D$.
- For $w \in W(A, D)$,
 - $H^-(A, w)$: nodes that have a center in L_{out} with type A .
 - $H^+(D, w)$: nodes that have a center in L_{in} with type D .
 - $A \rightsquigarrow D$ can be processed by i) retrieving $W(A, D)$, and ii) pairing every node in $H^-(A, w)$ with every node in $H^+(D, w)$, for each $w \in W(A, D)$.

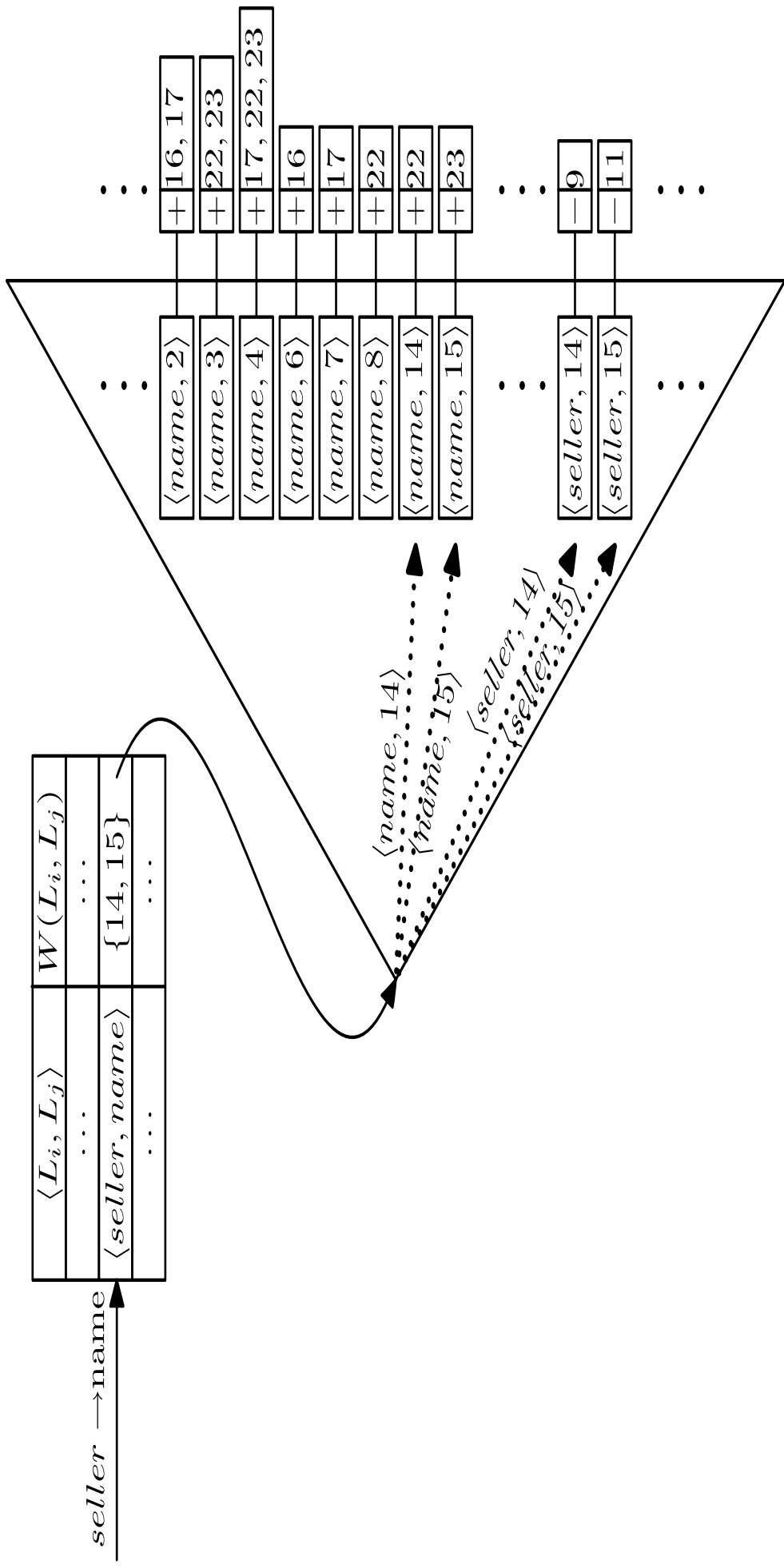
A B+-tree

- Let L_i be a type and w be a center.
- A B+-tree is built with the key (L_i, w) for searching $H^+(L_i, w)$ and $H^-(L_i, w)$.



An Example

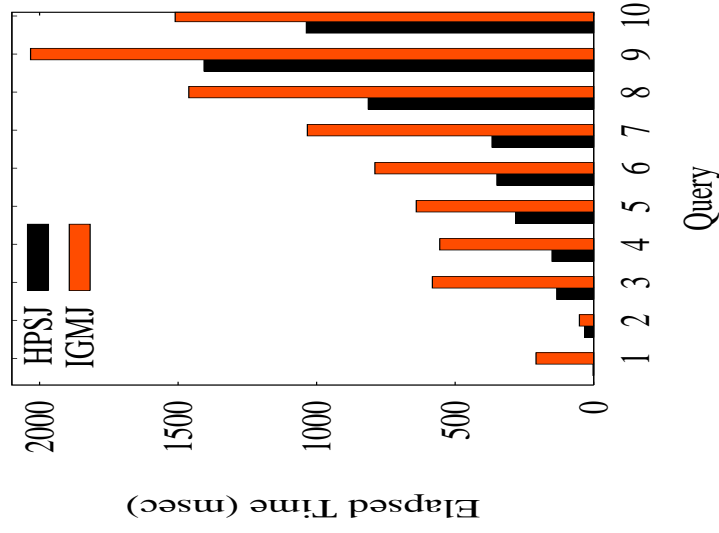
- Consider $seller \rightsquigarrow name$.



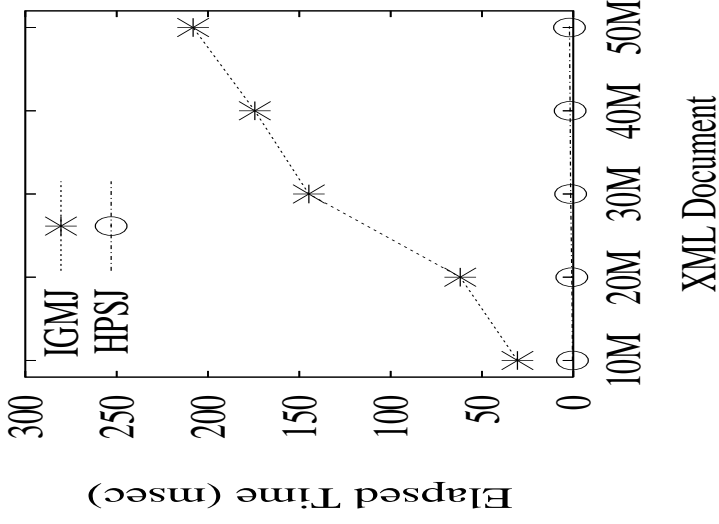
A Performance Study on Graphs

- XMark benchmark by treating both parent-child and ID/IDREF as edges in the same manner.
- Compare our 2-hop based labeling algorithm, *HPSJ*, with the interval based labeling algorithm, *IGMJ*.

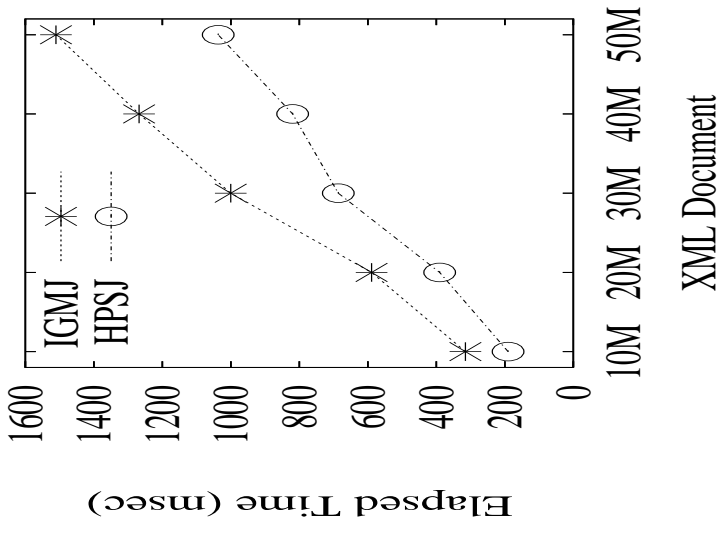
On Graphs



(a) 50M



(b) Q1



(c) Q10

- *HPSJ significantly outperforms IGMJ in all cases.*

Conclusion

- We proposed fast computing 2-hop labeling.
- We proposed a join-index to efficiently process reachability query processing of $A \rightsquigarrow D$.
- We showed that our approach can significantly outperform the up-to-date algorithm for reachability queries over graphs, and achieve high efficiency for processing reachability queries over trees.