EDBT: 15th International Conference on Extending Database Technology
Mar. 27, 2012 – Berlin, Germany

Shortest-Path Queries for Complex Networks: Exploiting Low Tree-width Outside the Core

Takuya Akiba
(The University of Tokyo)
Christian Sommer
(MIT CSAIL)
Ken-ichi Kawarabayashi
(National Institute of Informatics)

Introduction: Complex Networks

Real World Networks

- Social Networks
- Web Graphs
- Biological Networks
- Technological Networks

Synthetic Models

- Preferential Attachment
- Kronecker Graphs

“**Complex Networks**”

scale-free, small-world, core-fringe, …

Introduction: Shortest Paths on Networks

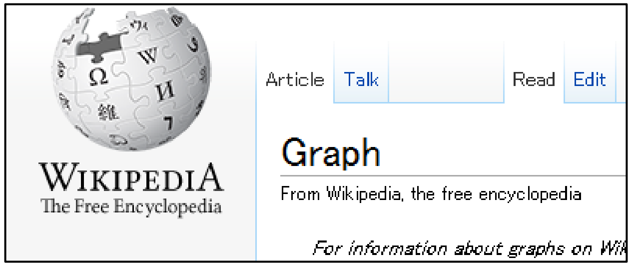
Context-Aware Search
Distance on web graphs

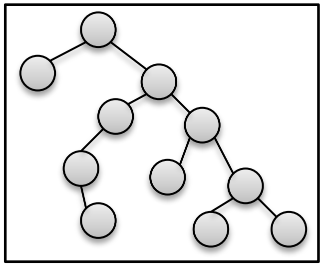
Socially-Sensitive Search
Distance on social networks

Social Network Analysis

Biological Analysis

⋮





Introduction: Shortest Path Queries

- **Trivial:** Breadth-First Search (BFS)
  - Too slow (for large networks & interactive situations)

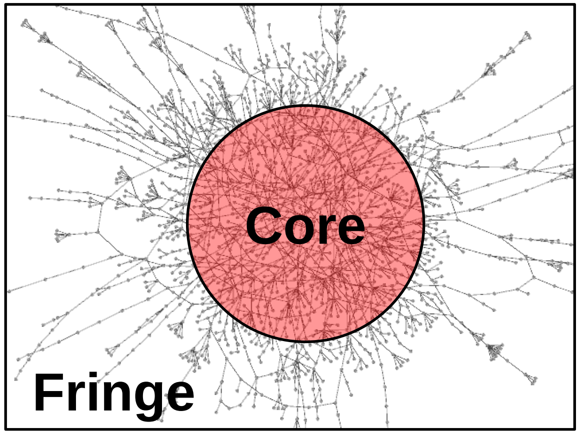
- **Solution:** Precomputing indices
  1. Precompute an index
  2. Answer queries using the index

Goal: **Good trade-off** between

- Indexing time
- Index size
- Query time
- Accuracy
  (for approximate methods)

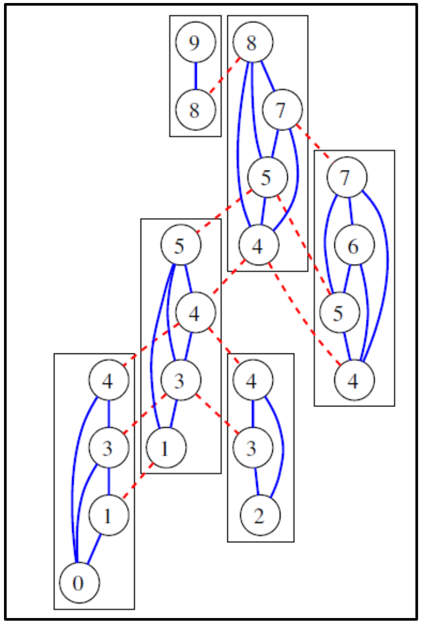
Core-Fringe Structure

of Complex Networks



Dense core + tree-like trails

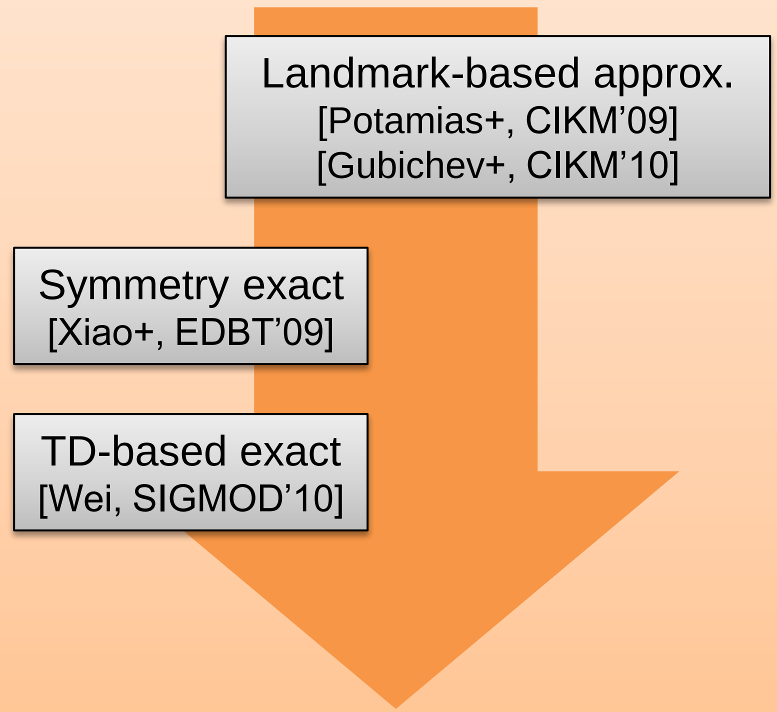
Tree Decompositions



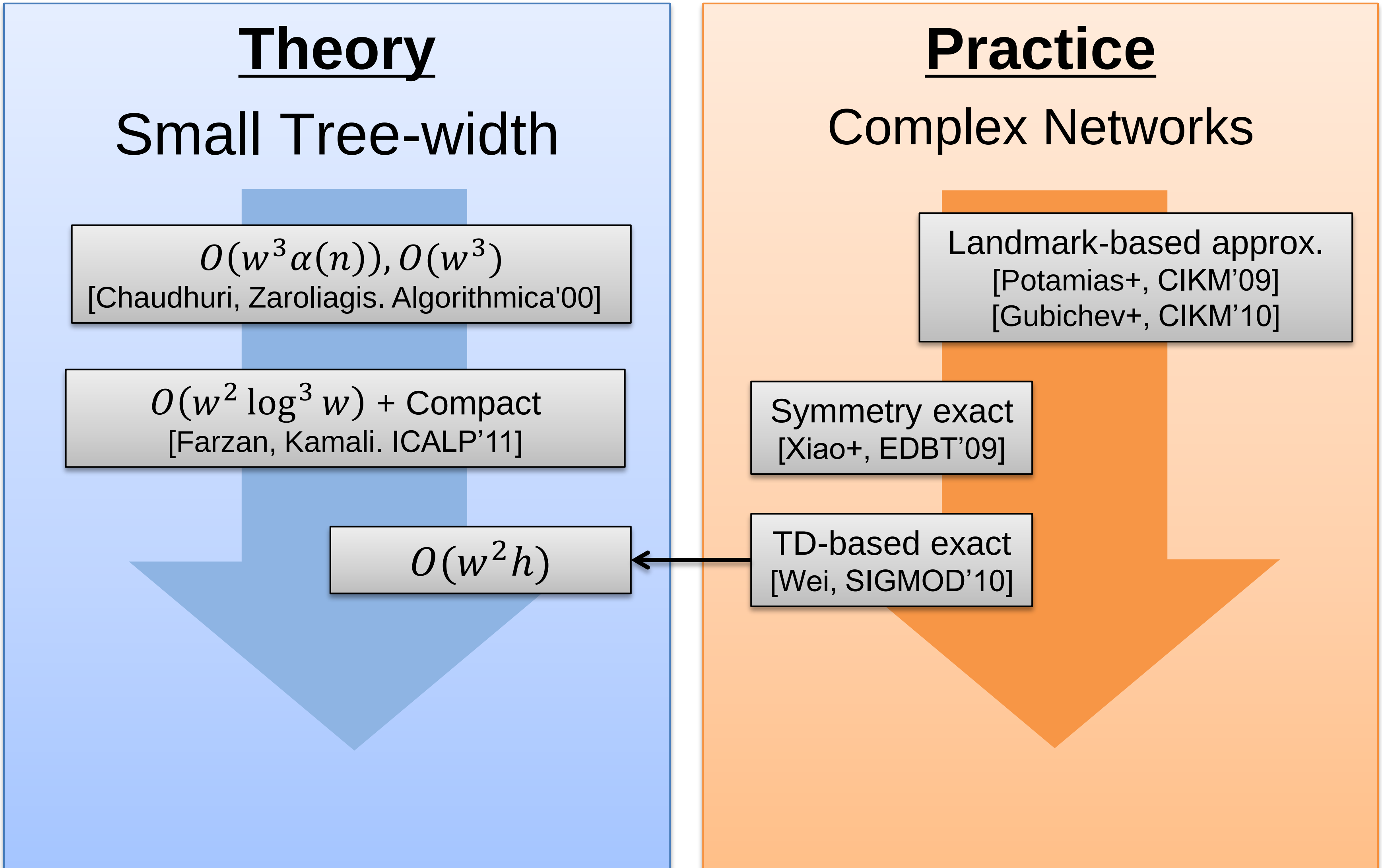
Introduction: History & Contribution

<u>**Practice**</u>

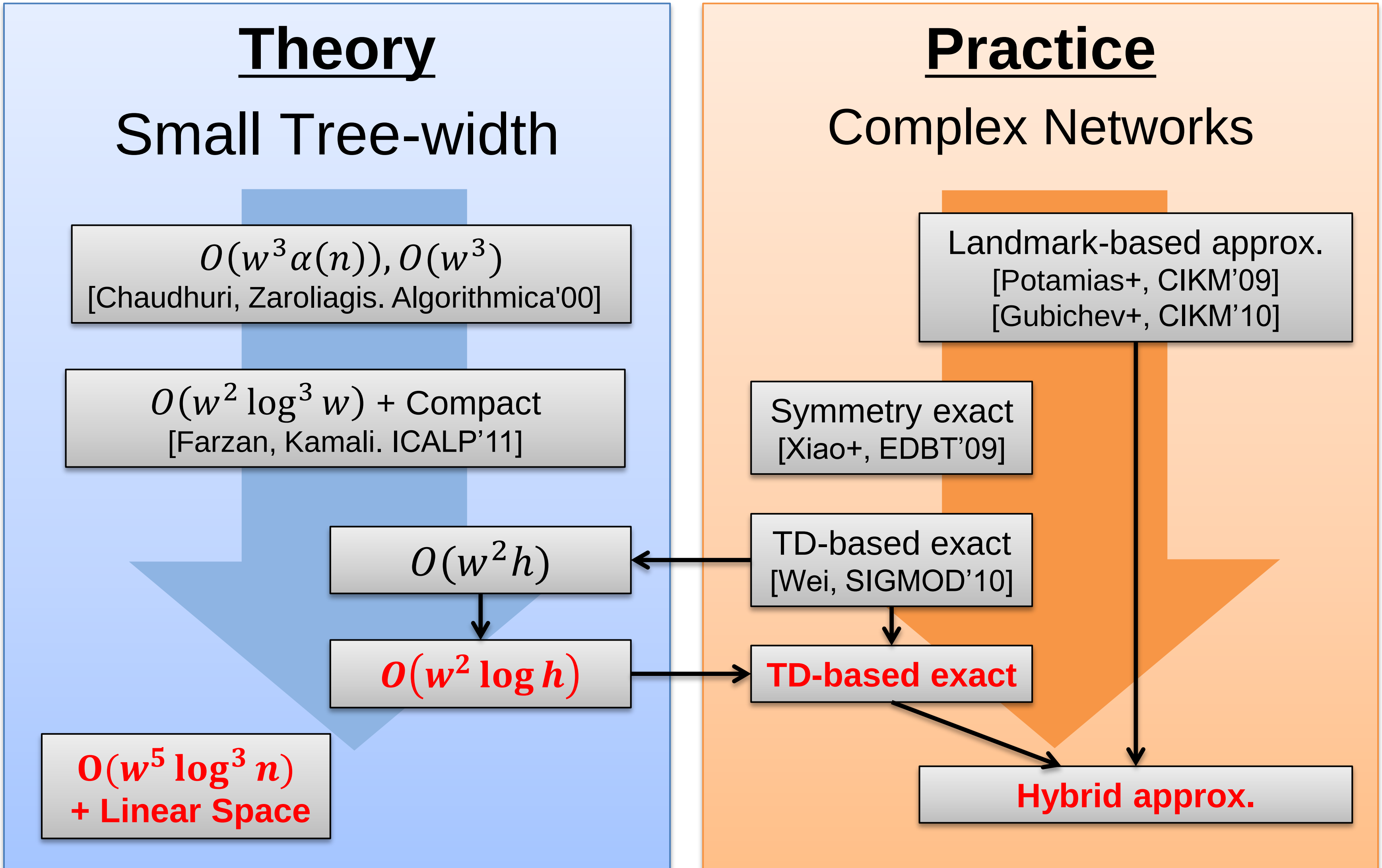
Complex Networks



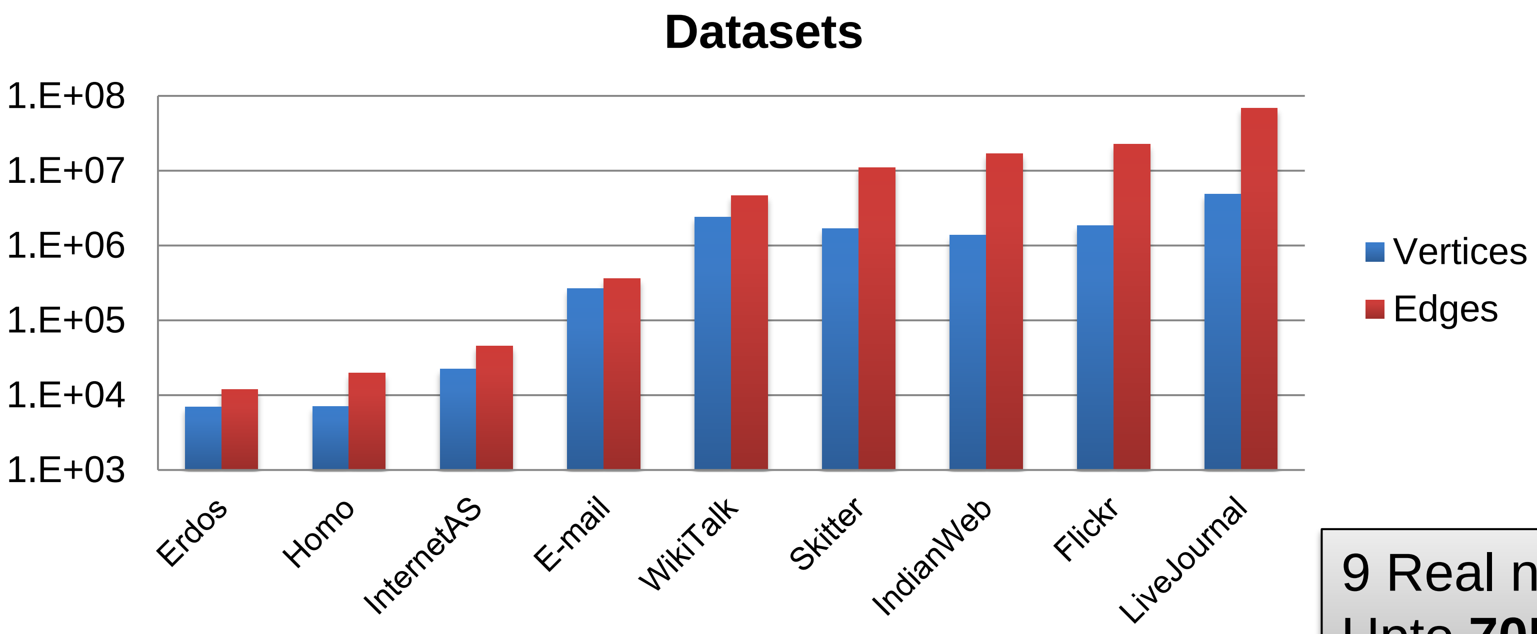
Introduction: History & Contribution



Introduction: History & Contribution



Introduction: Summary of Experiments



9 Real networks
Upto **70M** edges

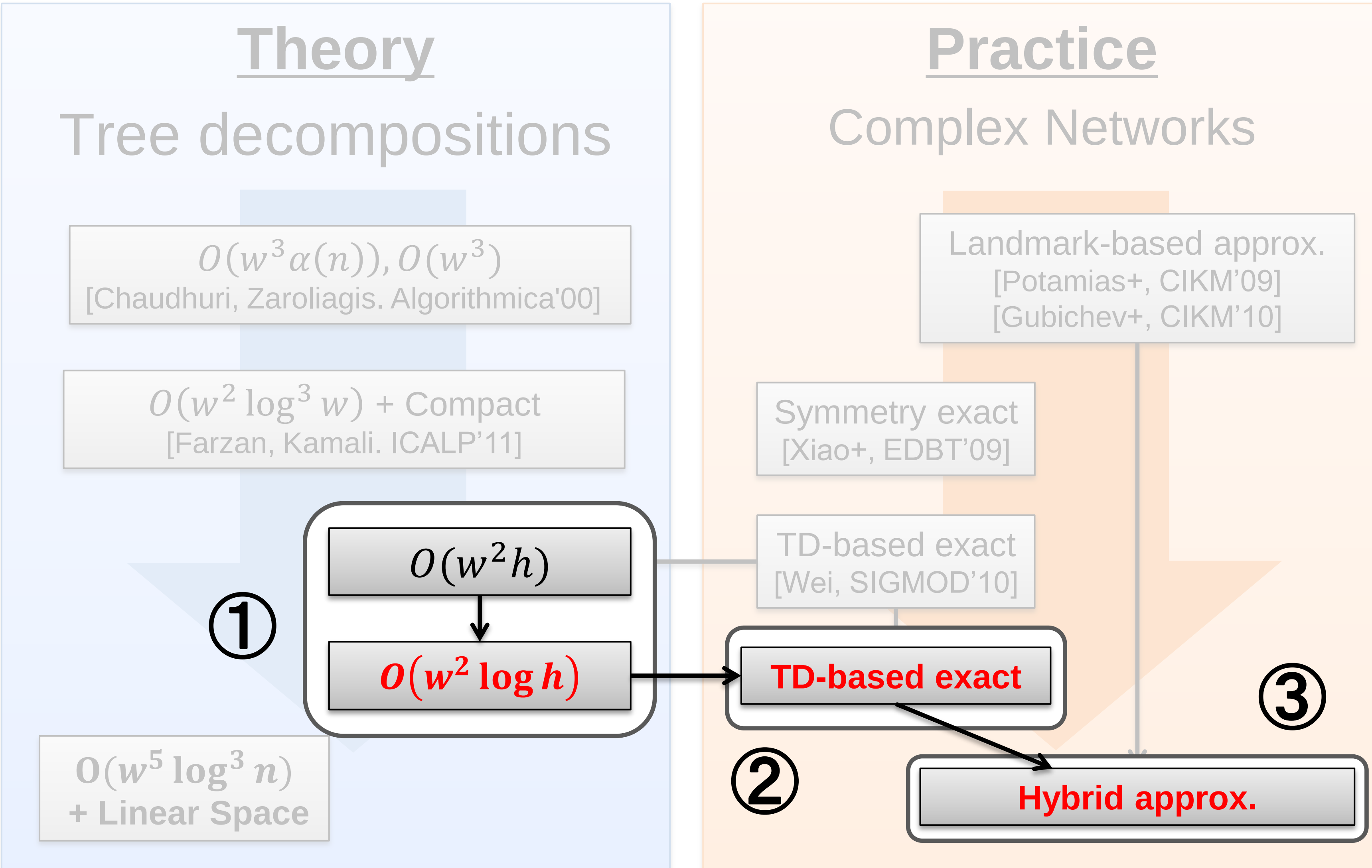
Exact TD-Based Method

- Upto 20x faster preprocessing + faster querying, data size, …

Approx. Hybrid Method

- Upto 2x smaller index size + better accuracy, …

Introduction: Outline

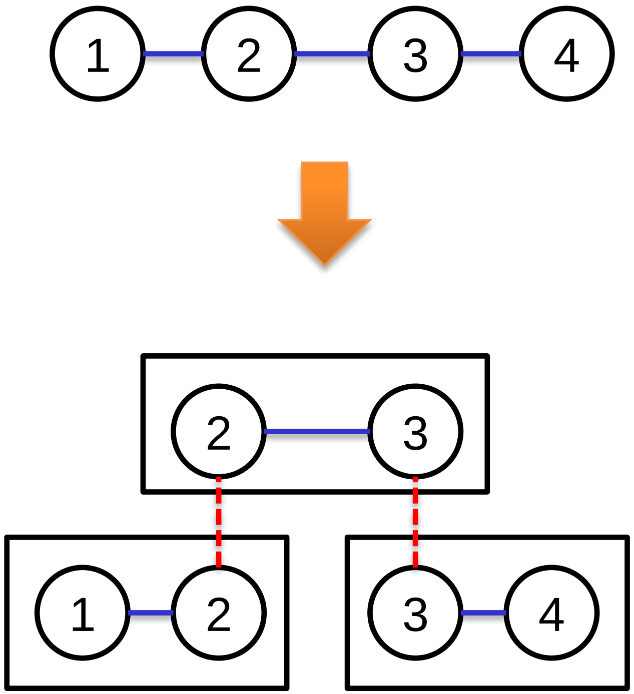


Theoretical Contribution

Query Processing for Graphs with Small Tree-Width

Tree Decomposition [Robertson, Seymour. '84]

Tool to treat **tree-like graphs** as **trees**



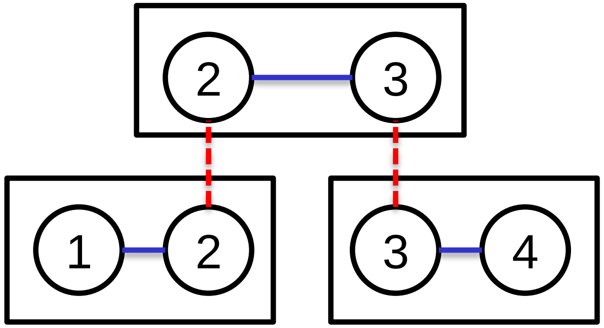
1. Every **vertex** appears at least once.
2. Every **edge** is contained in at least one bag
3. Every vertex induces a **subtree**

Width [Robertson, Seymour. '84]

The width of a tree-decomposition

≒How tree-like is it ?

Definition: **(Maximum bag size) - 1**

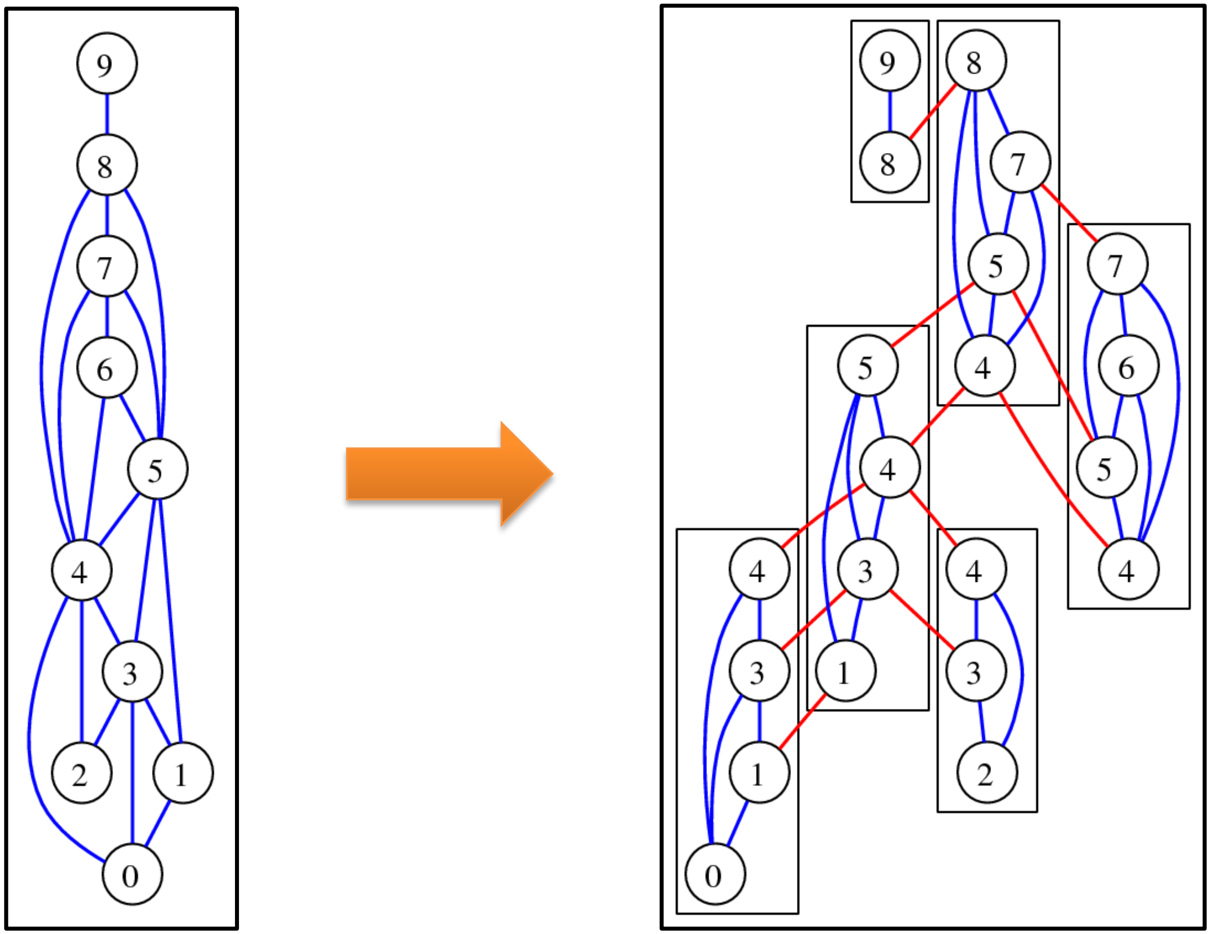


Maximum bag size = 2

Width = 1

Smaller → Tree-like → Easy

Another Example



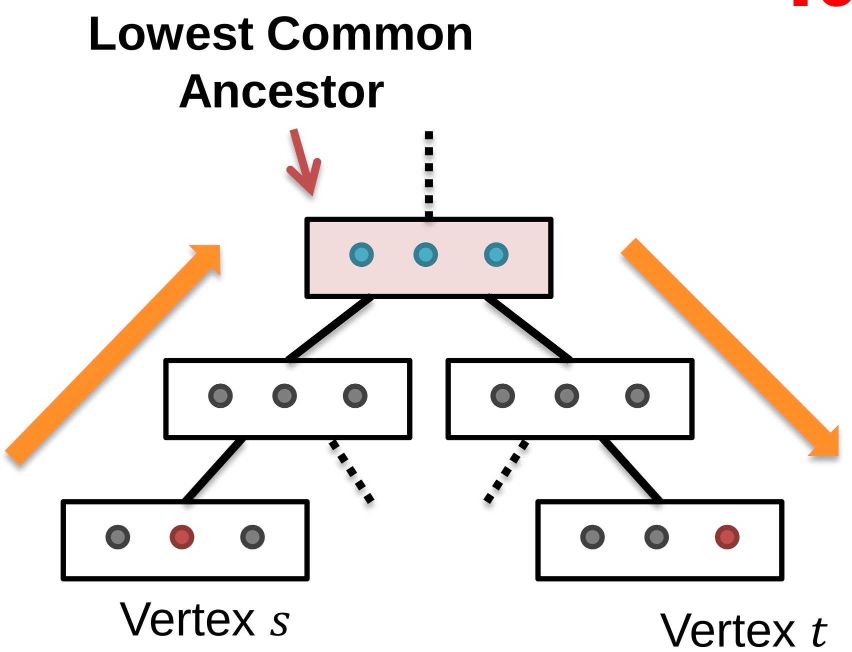
Width = 3

Methods for tree decompositions with width w

Literature	Space	Query Time	Comment
[CZ'00]	$O(w^3 n)$	$O(w^3 \alpha(n))$	α : inverse of Ackermann function
[CZ'00]	$O(w^3 n \log n)$	$O(w^3)$	
[FK'11]	$w(n + o(n) - w/2) + O(n)$	$O(w^2 \log^3 w)$	Unweighted, Undirected; Succinct
[Wei'10]	$O(w^2 b)$	$O(w^2 h)$	h : height of TD b : # of bags ($b = O(n)$)
Ours 1	$\boldsymbol{O(w^2 b)}$	$\boldsymbol{O(w^2 \log h)}$, $\boldsymbol{O(w^2 \log \log n)}$	
Ours 2	$\boldsymbol{O(m)}$	$\boldsymbol{O(w^5 \log^3 n)}$	**Linear space**

$O(w^2\ b)$ Space, $O(w^2\ h)$ Time [Wei, SIGMOD'10]

Idea



Every path passes LCA bag

Compute the distance:

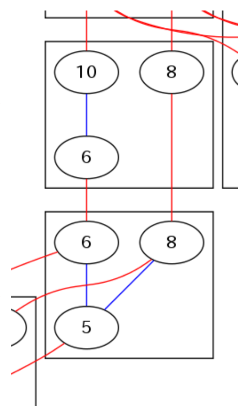
1. s to **every vertex in LCA**
2. **Every vertex in LCA** to t

$O(w^2\ b)$ Space, $O(w^2\ h)$ Time [Wei, SIGMOD'10]

Store **distance matrix** for each bag.

	6	8	10
6	0	2	1
8	2	0	2
10	1	2	0

	5	6	8
5	0	1	1
6	1	0	2
8	1	2	0



$$O(w^2) \times b$$

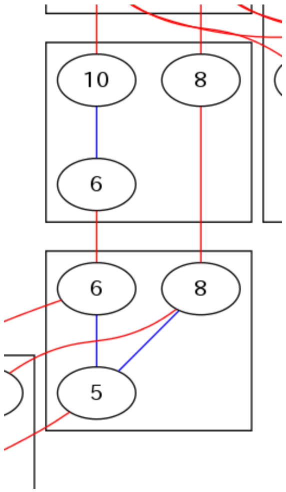
$$= \ O(w^2 b) \text{ Space}$$

$O(w^2\ b)$ Space, $O(w^2\ h)$ Time [Wei, SIGMOD'10]

Climbing
1 Step

V	6	8	10
Dist	2	3	3

V	5	6	8
Dist	2	2	3



	6	8	10
6	0	2	1
8	2	0	2
10	1	2	0

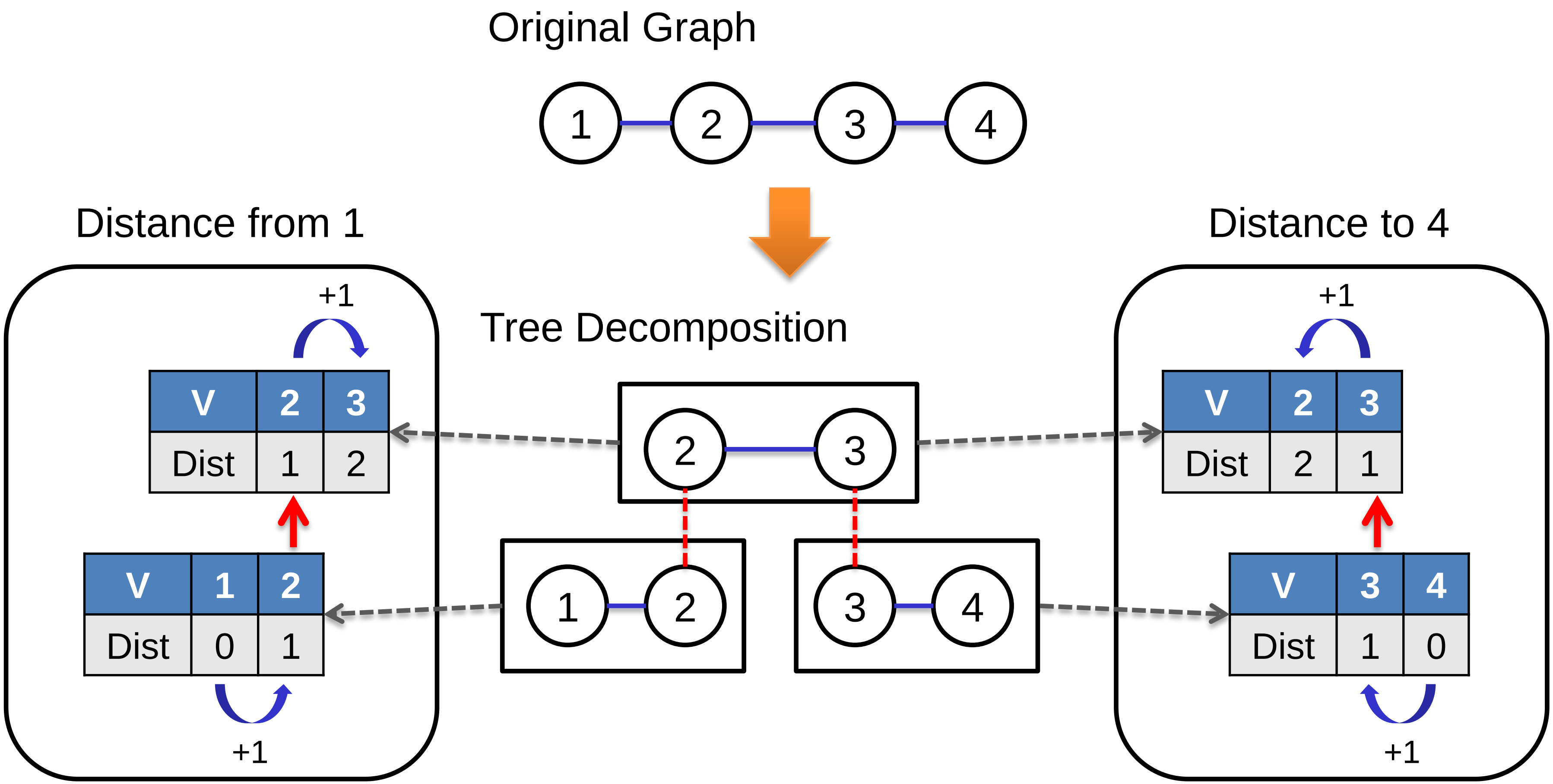
Distance matrix

Climb bags conducting **dynamic programming**

$$O(w^2) \times O(h) = O(w^2 h) \text{ Time}$$

1 step **Num of steps**

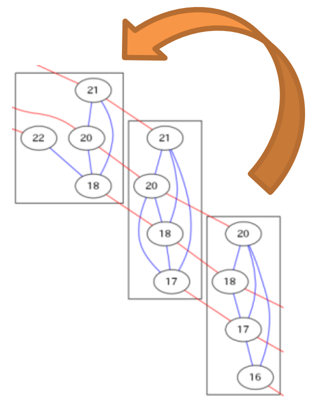
Example



$$d(1,4) = d(1,2) + d(2,4) = 1 + 2 = 3$$

$O(w^2\ b)$ Space, $O(w^2 \log h)$ Time

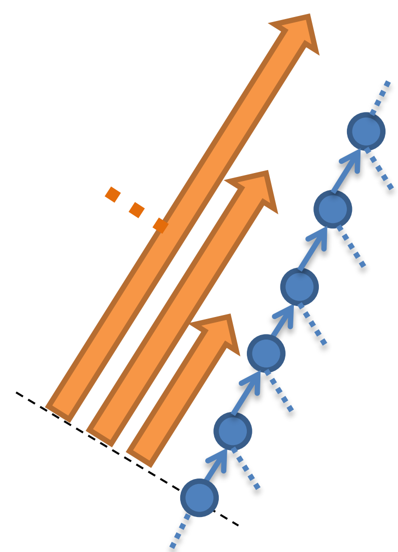
Idea



	18	20	21	22
16	2	1	2	3
17	1	1	1	2
18	0	1	1	1
20	1	0	1	2

Directly climb to 2^i th ancestor,
$O(w^2 \log h)$ query time

(We omit the detail)



Practical Contribution

Application to Complex Networks: Exact Method

Relaxed Tree Decompositions

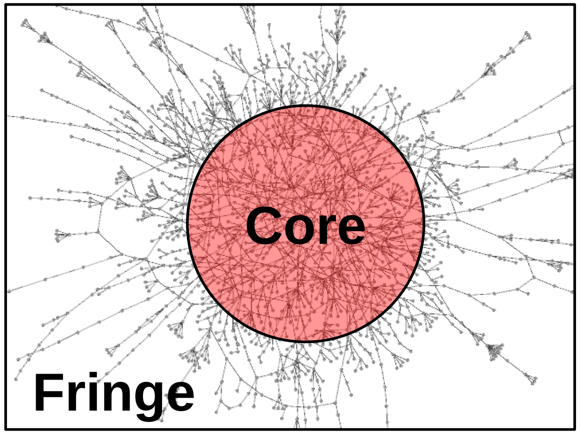
- No good tree decompositions for real networks
  - Tree decomposition is a tool for tree-like graphs
  - Complex networks are not tree-like

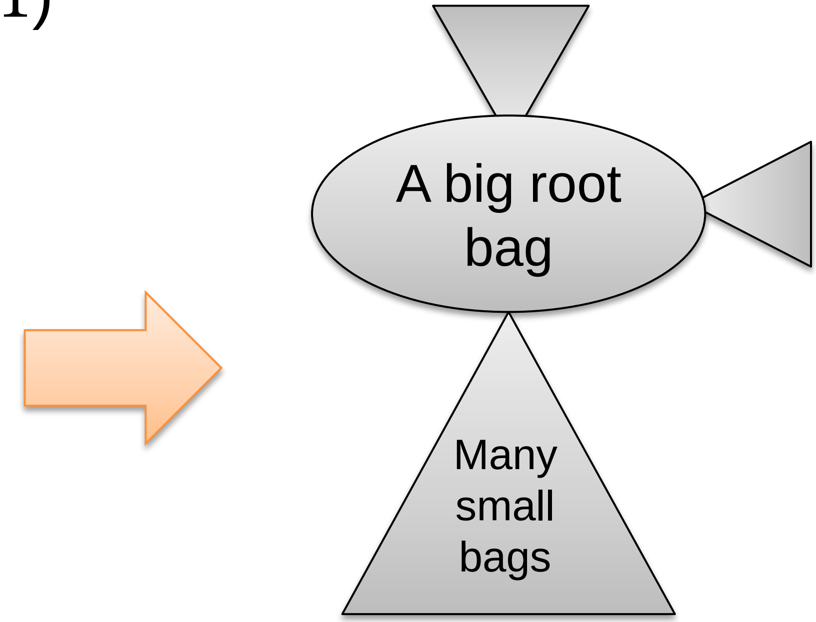
- However, they have core-fringe structure
  - **Dense** core + **tree-like** fringe

- **Idea**: Decomposing tree-like fringe using tree Decompositions

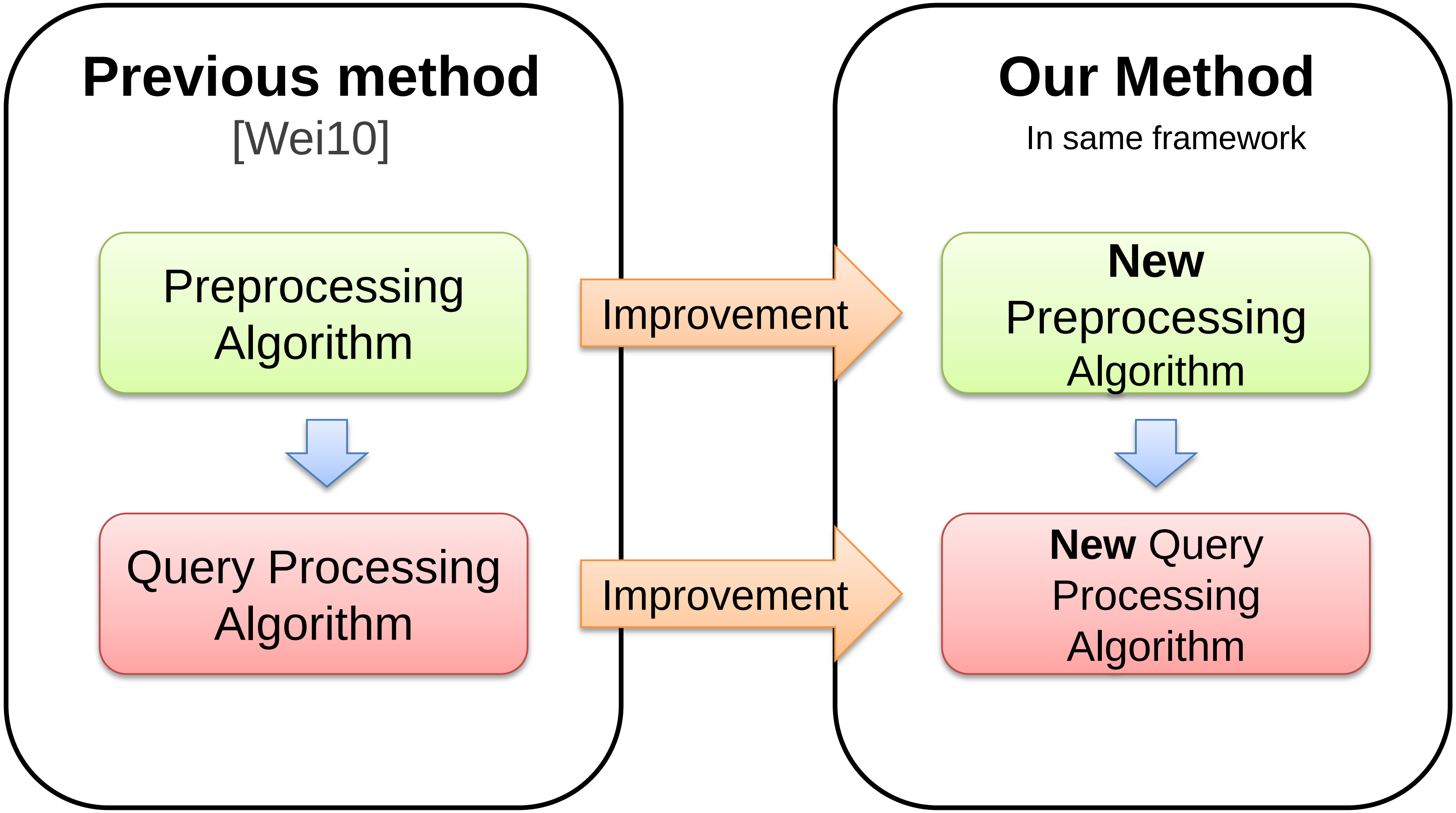
Relaxed Tree Decompositions

- Relaxed Tree Decomposition (relaxed width w)
  - One big bag for core
  - Many small bags for fringe
    (with size at most $w + 1$)





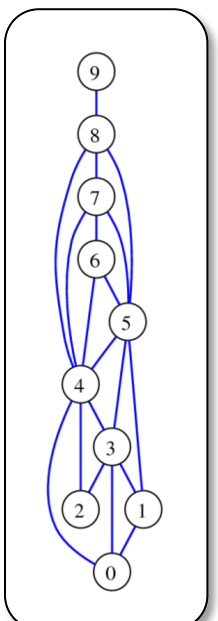
Our Method

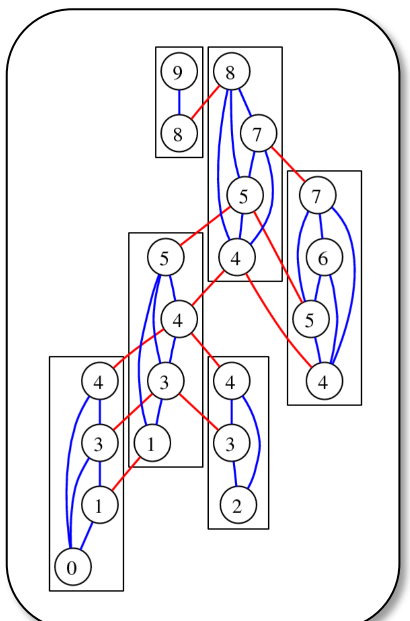


Previous method
[Wei10]
Preprocessing Algorithm
Query Processing Algorithm
Improvement
Improvement
Our Method
In same framework
New Preprocessing Algorithm
New Query Processing Algorithm

Preprocessing

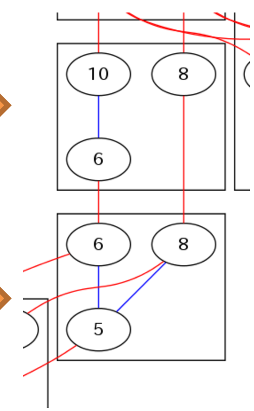
1. Tree decomposition heuristically
2. Shortest distance matrices





	6	8	10
6	0	2	1
8	2	0	2
10	1	2	0

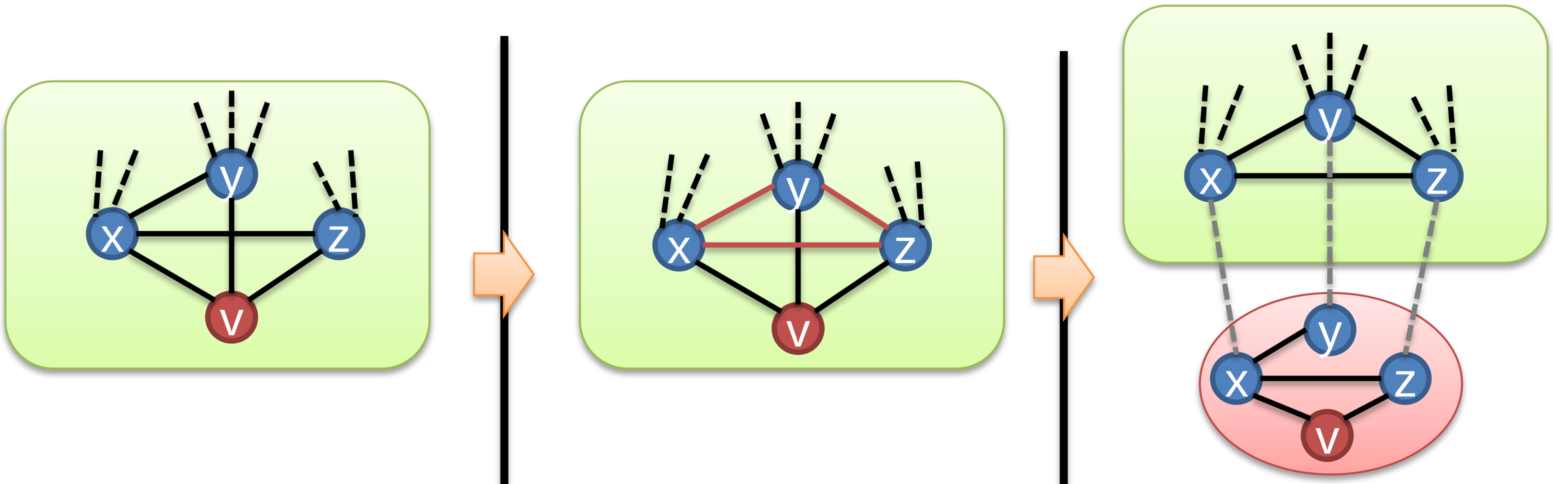
	5	6	8
5	0	1	1
6	1	0	2
8	1	2	0



Vertex Reduction

For any $v \in V \mid \deg(v) \leq w$

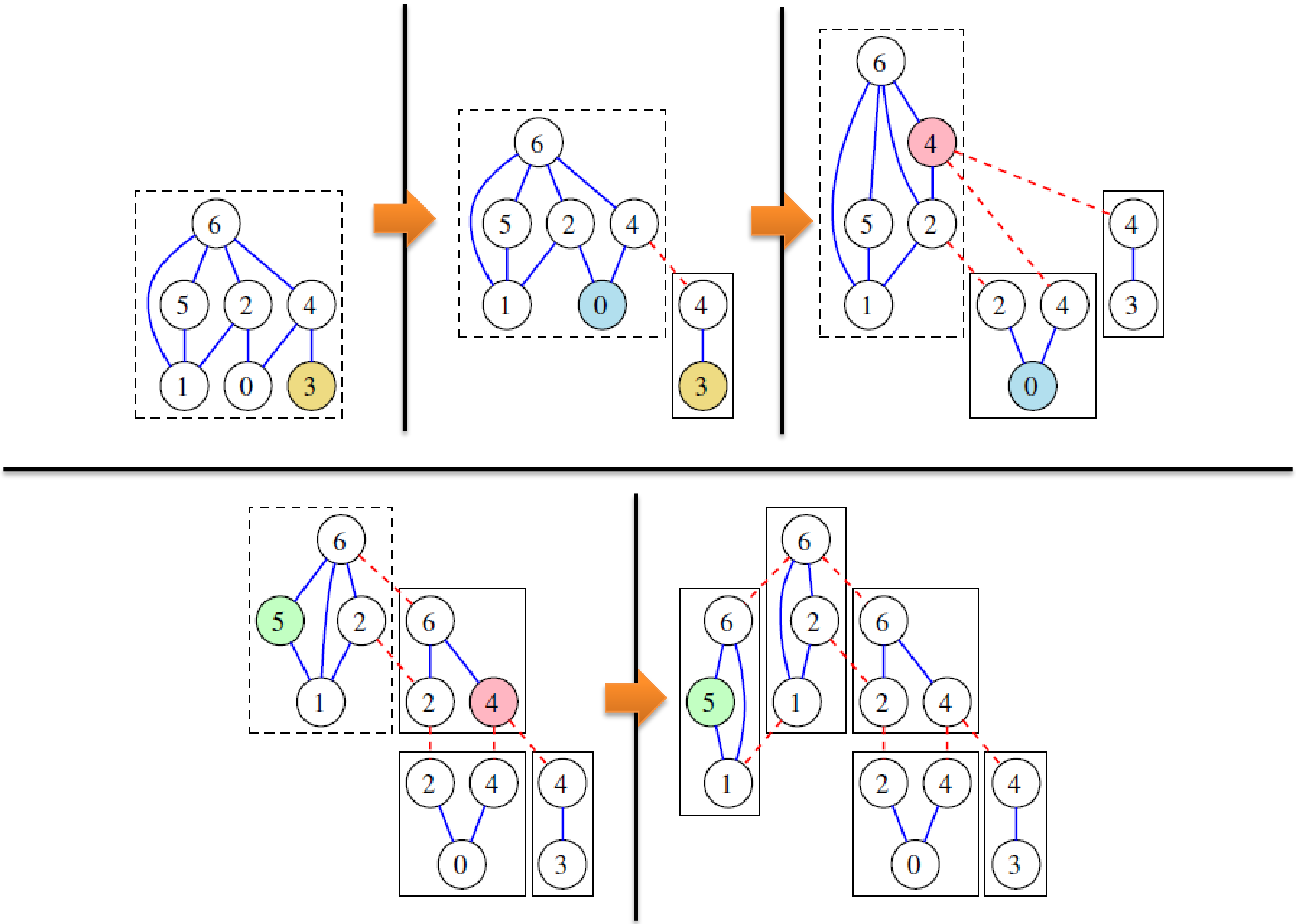
(w : parameter)



deg(v) = 3 | Add a clique | New bag

New bag size $\leq w + 1 \rightarrow$ Relaxed width w

Vertex Reduction



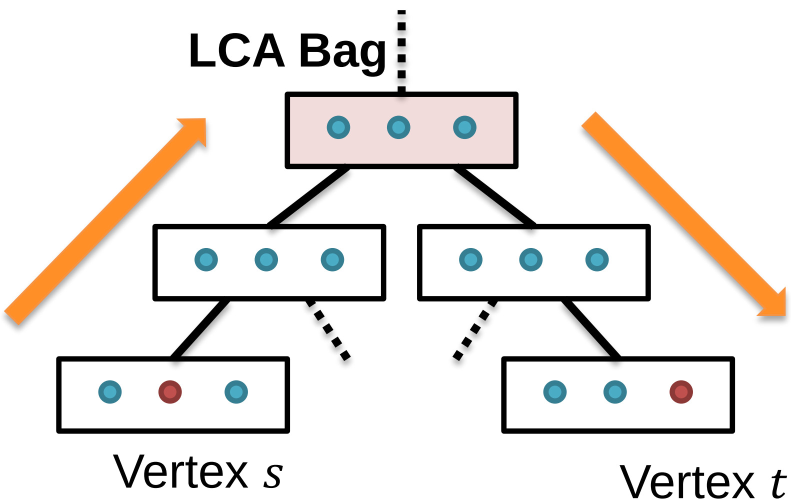
Shortest Distance Matrices

- Trivial: Compute them on original graph
- Our approach: Compute them on reduced graph
  - Reduced graphs are smaller
  - Though some vertices are deleted, actually we can compute all the matrices

SPQ method using TD

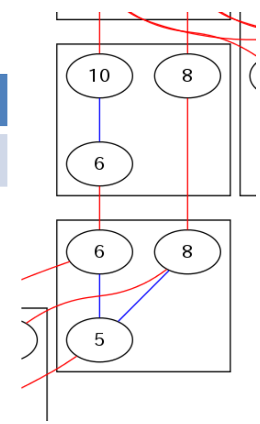
Query Processing

Dynamic programming climbing tree



V	6	8	10
Dist	2	3	3

V	5	6	8
Dist	2	2	3



Use improved algorithms from the first part

Practical Contribution

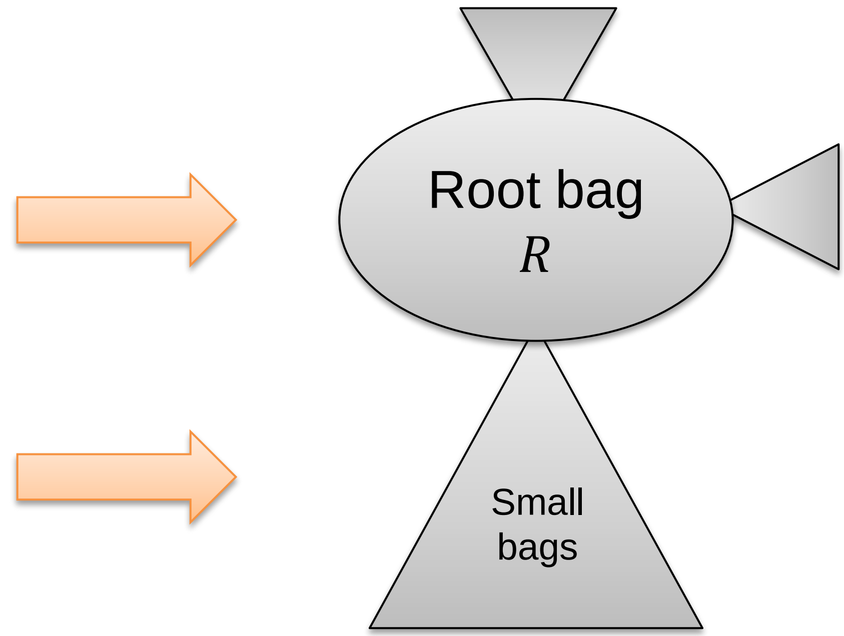
Application to Complex Networks: Hybrid Approximate Method

Hybrid Approximation Method

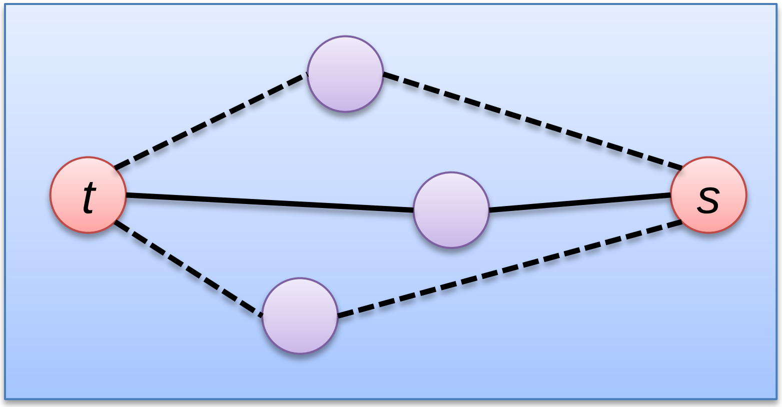
- Bottleneck of exact method: root bag R
  - $\Omega(|R|^2)$ time and space

Other existing method

Tree decomposition



Landmark-based Estimation [Potamias+, CIKM'09]

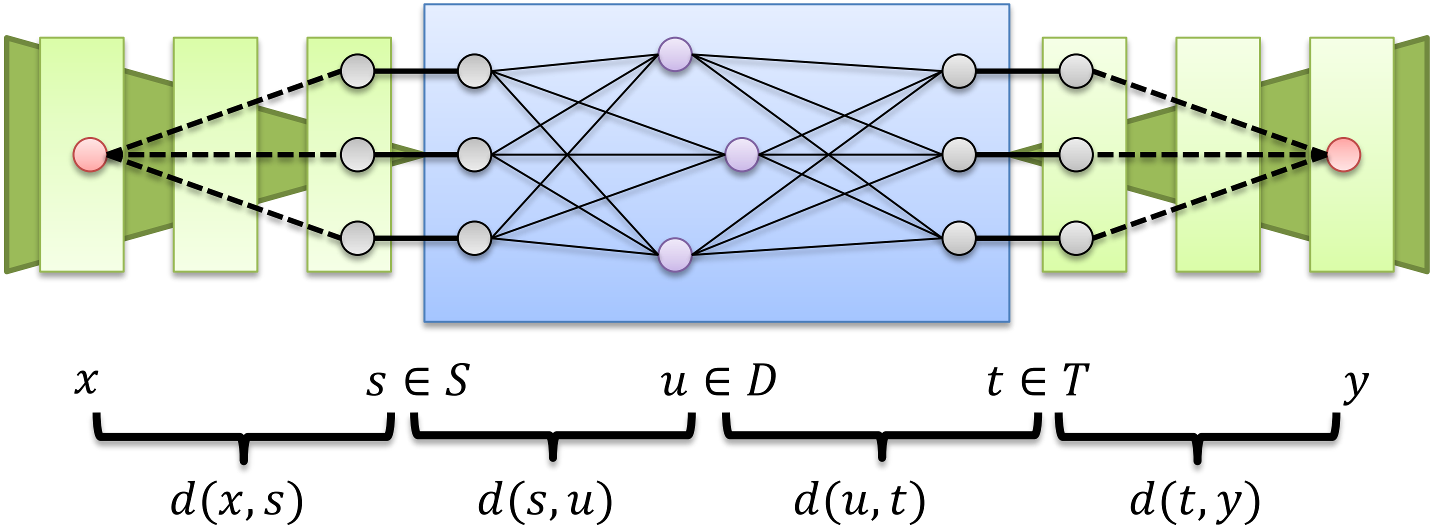


$$\widetilde{d_G}(s,t) = \min_{u \in D}\{ d_G(s,u) + d_G(u,t)\}$$

(Triangulation)

Simple and practical

Hybrid with landmark-based method



$$\tilde{d}(x,y)$$

$$= \min_{s \in S, t \in T} \{ d(x,s) + \tilde{d}(s,t) + d(t,y) \}$$

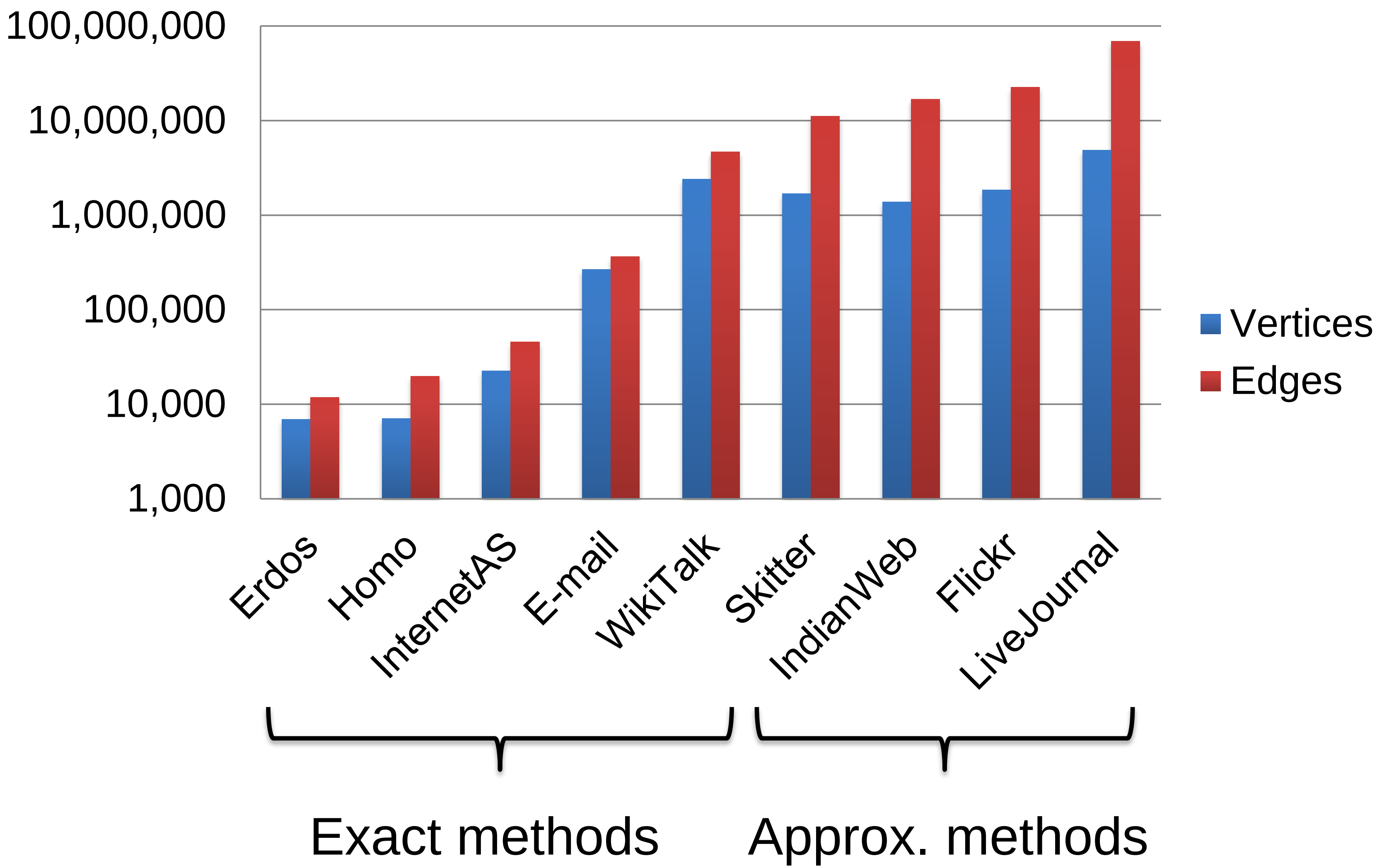
$O(w^2 h + w^2 |D|)$ time

$$= \min_{u \in D} \left\{ \min_{s \in S} \{ d(x,s) + d(s,u) \} + \min_{t \in T} \{ d(u,t) + d(t,y) \} \right\}$$

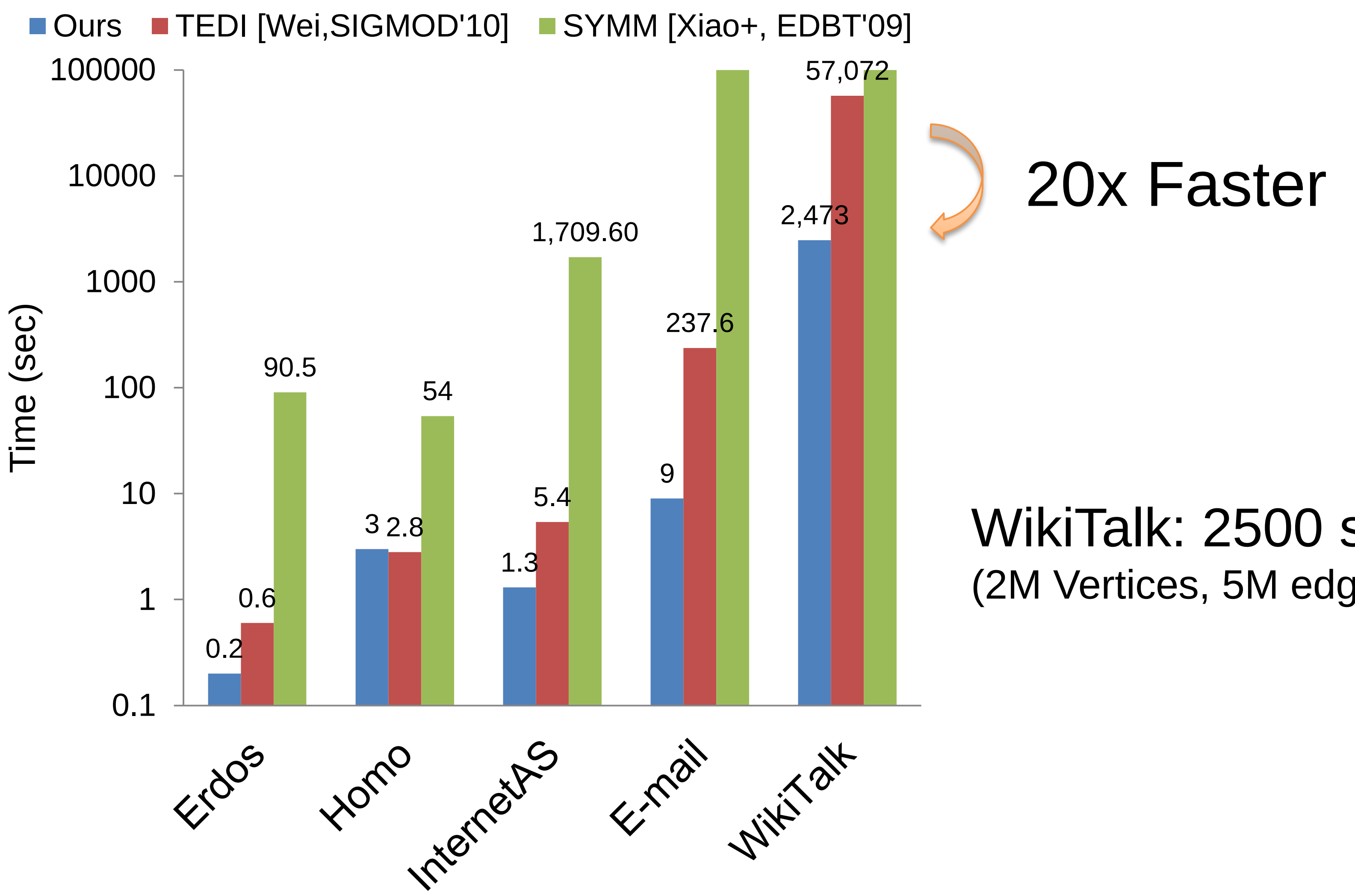
$O(w^2 h + w|D|)$ time

Experimental Evaluation

Real-World Datasets



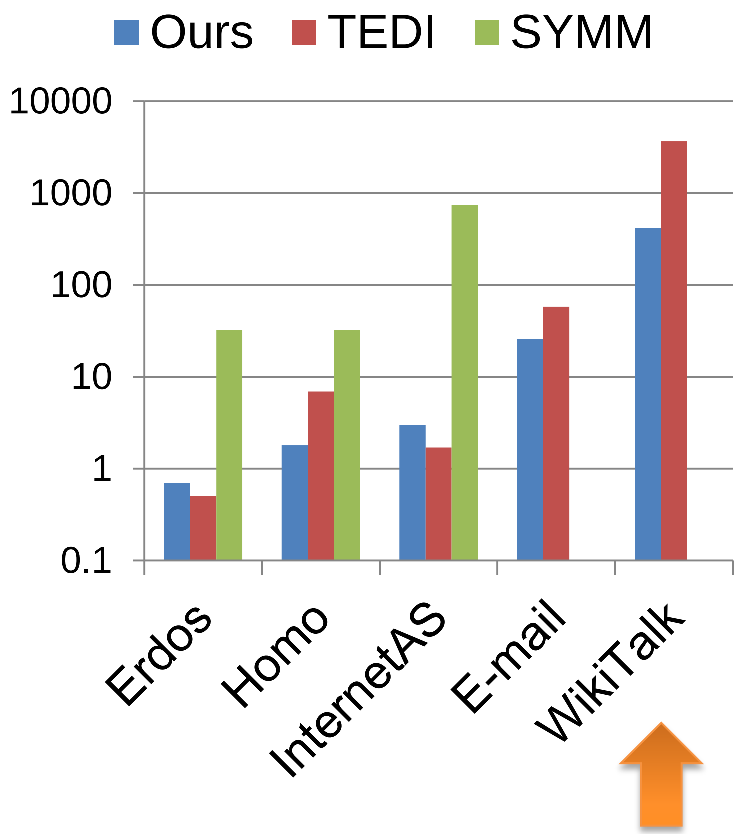
Exact Method: Preprocessing Time



WikiTalk: 2500 sec
(2M Vertices, 5M edges)

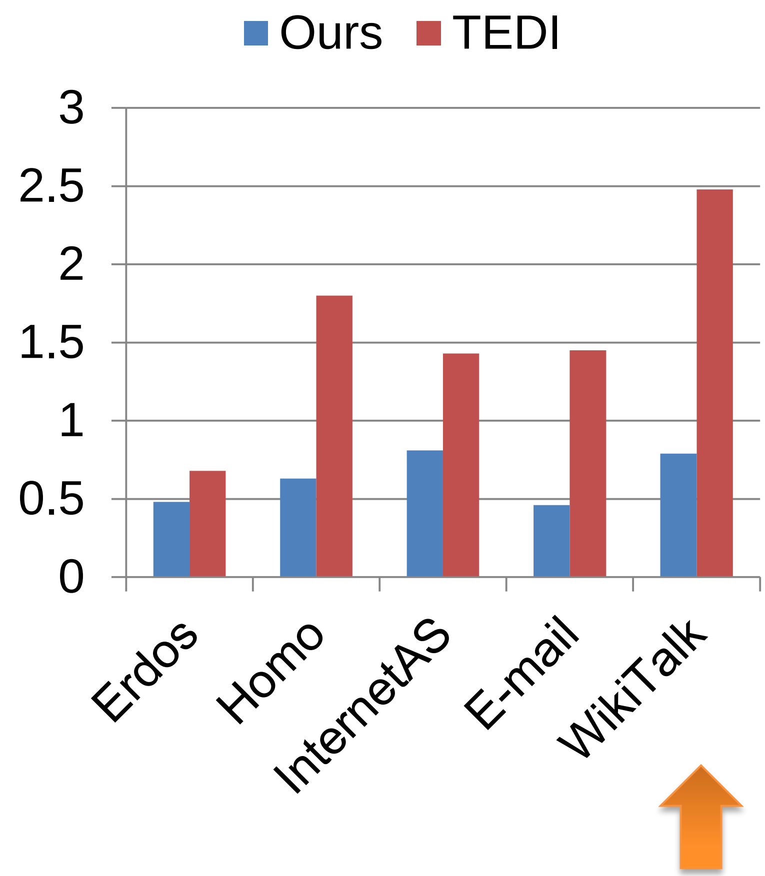
Exact Method: Index Size & Query TIme

Index Size (MB)



9x Smaller

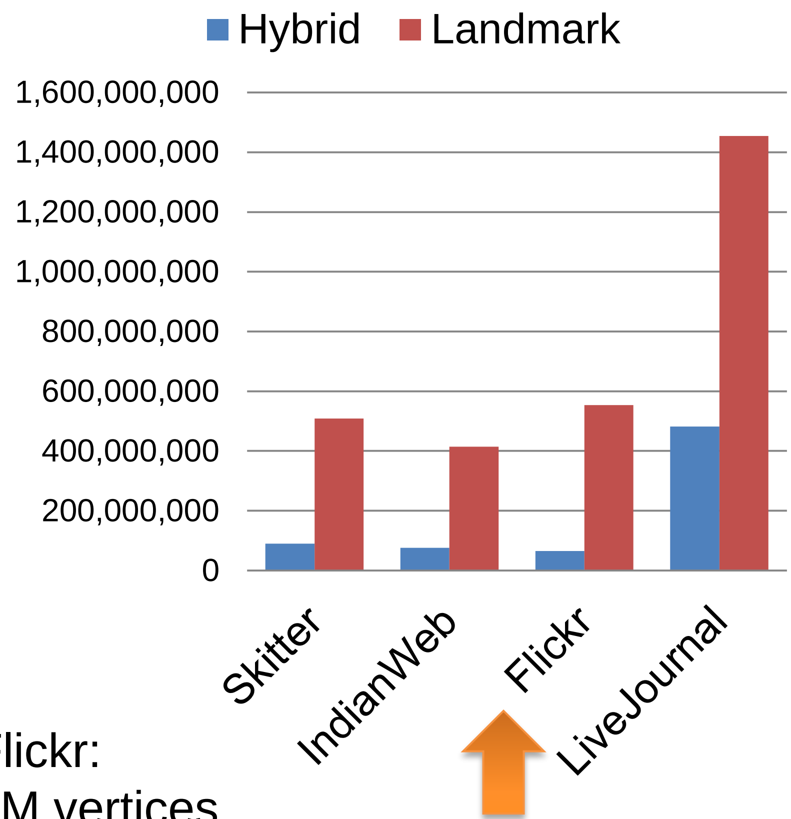
Query Time (μs)



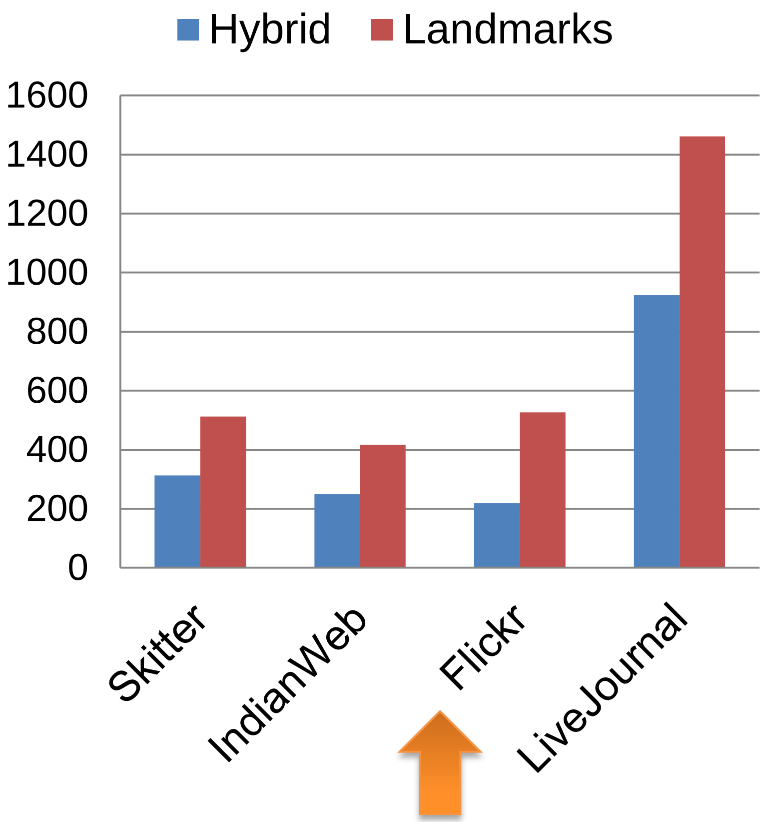
3x Faster

Approximate Method: Space

of Pairs
whose distance was stored



Index Size (MB)

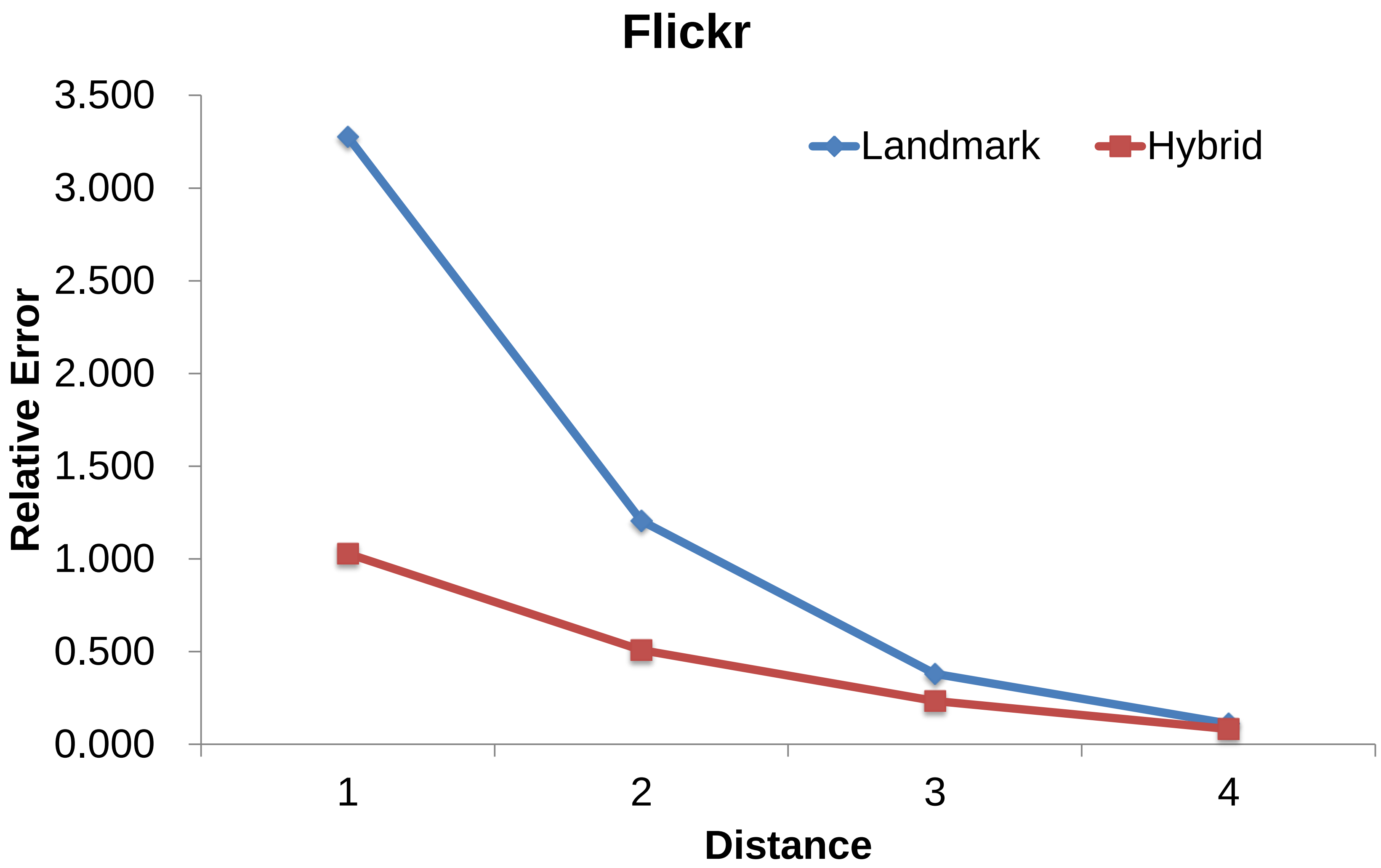


Flickr:
2M vertices,
23M edges

8x Less

2x Smaller

Approximate Method: Accuracy



Wrap Up

- Fast shortest path querying on large networks is useful in many applications
- Core-fringe structure of networks can be exploited by relaxed tree decompositions
- New exact method
  - With better preprocessing, query time and data size
- New hybrid approximate method
  - With better data size and accuracy

Core-fringe structure [Lu00]

Under the RPLG Model,

- $0 < \gamma < 2$
  - Dense "core" with diameter at most 3
  - "Tree-like trails" with constant length
- $2 < \gamma < 4$
  - Dense "core"
  - "Tree-like trails"
  - "Middle layer" between them
  - $O(\log n)$ path length