Pruned Labeling Algorithms:
Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries

Takuya Akiba (U Tokyo)

<Joint work with>
Yoichi Iwata (U Tokyo), Yuichi Yoshida (NII),
Yosuke Yano (U Tokyo), Yuki Kawata (U Tokyo),
Ken-ichi Kawarabayashi (NII)
Need for Graph Indexing Methods

Real-time Network-aware Services
(e.g., Network-aware Search)

Access\[\uparrow\]\[\downarrow\] Response

User

Distance between 1000 pair of vertices

In 10 ms (100 QPS)

Each distance should be computed in 10 μs

LSNA'14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Graph Indexing

Given a graph $G = (V, E)$

1. construct a data structure (an index) to
2. answer some queries quickly

Indexing

queries $(s_1, t_1), (s_2, t_2), ...$

Answer

Reachable!
Given a graph $G = (V, E)$

1. construct a data structure (an *index*) to
2. answer some *queries* quickly

**Goal:** Good trade-off

**Scalability**
- Indexing time
- Index size

**Query Performance**
- Query time
- Precision
Popular Graph Queries

Distance Queries on Social and Web Graphs

- Landmark [CIKM’09] [WSDM’10] [CIKM’10] [ICDE’12], TD [SIGMOD’10] [EDBT’12], 2-Hop [SIGMOD’12] [ESA’12], …

Distance Queries on Road Networks

- CH [WEA’08], CHASE [JEA’10], TNR [ALENEX’07], HL [SEA’11], …

Reachability Queries on Citation and XML Graphs

- GRAIL [VLDB’11], PWAH [SIGMOD’11], SCARAB [SIGMOD’12], TF-Label [SIGMOD’13], …
Popular Graph Queries

Distance Queries on Social and Web Graphs
- Landmark [CIKM’09] [WSDM’10] [CIKM’10] [ICDE’12], TD [SIGMOD’10] [EDBT’12], 2-Hop [SIGMOD’12] [ESA’12], …
- Pruned Landmark Labeling [SIGMOD’13]

Distance Queries on Road Networks
- CH [WEA’08], CHASE [JEA’10], TNR []

Reachability Queries on Citation and XML Graphs
- GRAIL [VLDB’11], PWAH [SIGMOD’11], SCARAB [SIGMOD’12], TF-Label [SIGMOD’13], …

Fast, Exact and Simple
Popular Graph Queries

Distance Queries on Social and Web Graphs
- Landmark [CIKM’09] [WSDM’10] [CIKM’10] [SIGMOD’10] [EDBT’12], 2-Hop [SIGMOD’12] [ESA’12], …
- Pruned Landmark Labeling [SIGMOD’13]

Distance Queries on Road Networks
- CH [WEA’08], CHASE [JEA’10], TNR [ALENEX’07], HL [SEA’11], …
- Pruned Highway Labeling [ALENEX’14]

Reachability Queries on Citation and XML Graphs
- GRAIL [VLDB’11], PWAH [SIGMOD’11], SCARAB [SIGMOD’12], TF-Label [SIGMOD’13], …
- Pruned Labeling with Landmarks and Paths [CIKM’13]

General Unified Approach
Popular Graph Queries

Distance Queries on Social and Web Graphs

• Landmark [CIKM’09] [WSDM’10] [CIKM’10] [ICDE’12], TD [SIGMOD’10] [EDBT’12], 2-Hop [SIGMOD’12] [ESA’12], …
• Pruned Landmark Labeling [SIGMOD’13]

Distance Queries on Road Networks

• CH [WEA’08], CHASE [JEA’10], TNR [ALENEX’07]
• Pruned Highway Labeling [ALENEX’14]

Reachability Queries on Citation and XML Graphs

• GRAIL [VLDB’11], PWAH [SIGMOD’11], SCARAB [SIGMOD’12], TF-Label [SIGMOD’13], …
• Pruned Labeling with Landmarks and Paths [CIKM’13]
1. Distance Queries on Complex Networks & Pruned Landmark Labeling [SIGMOD’13]

2. Extensions
   - Pruned Path Labeling [CIKM’13]
   - Pruned Highway Labeling [ALENEX’14]
   - Historical Pruned Landmark Labeling [WWW’14]
Distance Queries on Complex Networks & Pruned Landmark Labeling [SIGMOD’13]
Problem Definition

Given a graph $G = (V, E)$

1. construct an index to
2. answer distance $d_G(s, t)$

Goal: Good trade-off

Scalability
- Indexing time
- Index size

Query Performance
- Query time
- Precision
Real-world Networks

Road Networks

Complex Networks

- Social Networks
- Web Graphs
- Computer Networks
- Biological Networks

LSNA'14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Real-world Networks

Road Networks

Complex Networks

Structural differences → Different methods

LSNA'14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Real-world Networks

Road Networks

Complex Networks

First topic: methods for complex networks
Approx. Methods

Landmark-based methods
[Potamias+, CIKM’09]

Distance Sketch [Sarma+, WSDM’10]

Path Sketch [Gubichev+, CIKM’10]

Landmark-based method + LCA, Shortcuts, ...
[Qiao+, ICDE’12]
[Tretyakov+, CIKM’12]

MD

Highway 2-Hop
[Cheng+, EDBT’09]

Hierarchical HL
[Abraham+, ESA’12]

Pruned Labeling
(this work)

TD-based methods
[Wei, SIGMOD’10]

TD-based
[Akiba+, EDBT’12]

2-Hop Cover
[Cohen+, SODA’02]

2-Hop Cover
[Cheng+, EDBT’09]

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Approx. Methods

- Landmark-based methods
  - [Potamias+, CIKM’09]
- Distance Sketch
  - [Sarma+, WSDM’10]
- Path Sketch
  - [Gubichev+, CIKM’10]

Landmark-based method + LCA, Shortcuts, ...
  - [Qiao+, ICDE’12]
  - [Tretyakov+, CIKM’12]

Hierarchical HL
  - [Abraham+, ESA’12]

Highway 2-Hop
  - [Jin+, SIGMOD’12]

2-Hop Cover
  - [Cohen+, SODA’02]

2-Hop Cover (Main)
  - [Cheng+, EDBT’09]

TD-based
  - [Akiba+, EDBT’12]

TD-based
  - [Wei, SIGMOD’10]

Pruned Labeling
  - (this work)

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Approx. Methods

- Landmark-based methods
  - [Potamias+ , CIKM’09]

- Distance Sketch
  - [Sarma+, WSDM’10]

- Path Sketch
  - [Gubichev+, CIKM’10]

- Landmark-based method + LCA, Shortcuts, ...
  - [Qiao+, ICDE’12]
  - [Tretyakov+, CIKM’12]

Exact Methods

- 2-Hop Cover
  - [Cohen+, SODA’02]

- 2-Hop Cover
  - [Cheng+, EDBT’09]

- TD-based
  - [Wei, SIGMOD’10]

- Highway 2-Hop
  - [Jin+, SIGMOD’12]

- TD-based
  - [Akiba+, EDBT’12]

- Hierarchical HL
  - [Abraham+, ESA’12]

- Pruned Labeling
  - (this work)

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Landmark-based Method [Potamias+, CIKM’09]

Single landmark version

1. Precomputation
   - Select a vertex \( l \) as a landmark
   - Compute \( d_G(v, l) \) for every vertex \( v \)

2. Query
   \[
   \overline{d}_G(s, t) = d_G(s, l) + d_G(l, t)
   \]
Multiple landmarks version:

1. Precomputation
   - Select a number (say 100) of vertices $D$ as landmarks
   - For each $u \in D$
     - Precompute $d_G(u, v)$ for every vertex

2. Query
   \[
   \overline{d}_G(s, t) = \min_{u \in D} \{ d_G(s, u) + d_G(u, t) \} 
   \]
Key Factor for Good Average Precision

**Hubs in Real Networks**

- Exceptionally central vertices
  - High degree (power-law degree distribution)

- Large portion of shortest paths pass nearby these vertices

- Select these vertices as landmarks
Precision Problem

- Total average precision is nice
- Precision for close pairs is terrible
  - [Akiba+, EDBT’12], [Qiao+, ICDE’12]
Methods

Exact Methods

- 2-Hop Cover
  - [Cohen+, SODA’02]
- Highway 2-Hop
  - [Jin+, SIGMOD’12]
- Hierarchical HL
  - [Abraham+, ESA’12]
- TD-based
  - [Wei, SIGMOD’10]
- TD-based
  - [Akiba+, EDBT’12]

Approx. Methods

- Landmark-based methods
  - [Potamias+, CIKM’09]
- Distance Sketch
  - [Sarma+, WSDM’10]
- Path Sketch
  - [Gubichev+, CIKM’10]
- Landmark-based method
  + LCA, Shortcuts, ...
    - [Qiao+, ICDE’12]
    - [Tretyakov+, CIKM’12]

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Approx. Methods

- Landmark-based methods
  - [Potamias+, CIKM’09]  

- Path Sketch
  - [Gubichev+, CIKM’10]  

- Bad precision for close pairs
  - WSDM’10

- Slow querying (ms)

**Exact Methods**

- 2-Hop Cover
  - [Cohen+, SODA’02]  

- 2-Hop Cover
  - [Cheng+, EDBT’09]  

- Highway 2-Hop
  - [Jin+, SIGMOD’12]  

- TD-based
  - [Wei, SIGMOD’10]  

- TD-based
  - [Akiba+, EDBT’12]  

- Hierarchical HL
  - [Abraham+, ESA’12]  

- Landmark-based method + LCA, Shortcuts, …
  - [Qiao+, ICDE’12]
  - [Tretyakov+, CIKM’12]  

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Approx. Methods

1. Landmark-based method + LCA, Shortcuts, ...
   [Qiao+, ICDE’12]
   [Tretyakov+, CIKM’12]

2. Distance Sketch [Sarma+, WSDM’10]

3. Path Sketch [Gubichev+, CIKM’10]

Exact Methods

1. TD-based [Wei, SIGMOD’10]

2. 2-Hop Cover [Cohen+, SODA’02]

3. 2-Hop Cover [Cheng+, EDBT’09]

4. Highway 2-Hop [Jin+, SIGMOD’12]

5. Hierarchical HL [Abraham+, ESA’12]

6. TD-based [Akiba+, EDBT’12]

7. Landmark-based methods [Potamias+, CIKM’09]

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Complex Networks roughly consist of two parts:

- Well-connected “Core”
- Tree-like “Fringes”
Tree-decomposition-based Approach
[Wei, SIGMOD’10][Akiba+, EDBT’12]

Core-Fringe Structure
of Complex Networks

Dense core + tree-like trails

Tree Decompositions
Tree Decomposition & Tree-width

[Seymour’84]

- Rather a tool of TCS
- Graph $\rightarrow$ Tree of vertex sets
- For tree-like graphs
  (Formally, graphs with small tree-width)
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 $w + 1$)
Preprocessing

1. Tree decomposition heuristically
2. Shortest distance matrices
Query Processing

Dynamic programming climbing tree
**Methods**

### Exact Methods
- 2-Hop Cover
  - [Cohen+, SODA’02]
- 2-Hop Cover
  - [Cheng+, EDBT’09]
- TD-based
  - [Wei, SIGMOD’10]
- Highway 2-Hop
  - [Jin+, SIGMOD’12]
- TD-based
  - [Akiba+, EDBT’12]
- Hierarchical HL
  - [Abraham+, ESA’12]

### Approx. Methods
- Landmark-based methods
  - [Potamias+, CIKM’09]
- Bad precision for close pairs
- Path Sketch
  - [Gubichev+, CIKM’10]
- Landmark-based method
  - + LCA, Shortcuts, ...
    - [Qiao+, ICDE’12]
    - [Tretyakov+, CIKM’12]
- Slow querying (ms)

---

SIGMOD Research 10, Akiba+: Fast Exact Shortest-Path Distance Queries on Large Networks by Pruned Landmark Labeling
Approx. Methods

- Landmark-based methods
  - [Potamias+, CIKM’09]

- Path Sketch
  - [Gubichev+, CIKM’10]

- Landmark-based method + LCA, Shortcuts, ...
  - [Qiao+, ICDE’12]
  - [Tretyakov+, CIKM’12]

Bad precision for close pairs

Cannot handle networks with ≥10M edges

2-Hop Cover
- [Cohen+, SODA’02]
- [Cheng+, EDBT’09]

TD-based
- [Wei, SIGMOD’10]
- [Akiba+, EDBT’12]

Hierarchical HL
- [Abraham+, ESA’12]

2-Hop Cover
- [Cheng+, EDBT’09]

Path Sketch
- [Gubichev+, CIKM’10]

Slow querying (ms)
Previous Methods

**Exact Methods**
- 2-Hop Cover
  - [Cohen+, SODA’02]
- 2-Hop Cover
  - [Cheng+, EDBT’09]
- TD-based
  - [Wei, SIGMOD’10]
- Highway 2-Hop
  - [Jin+, SIGMOD’12]
- TD-based
  - [Akiba+, EDBT’12]
- Hierarchical HL
  - [Abraham+, ESA’12]

**Approx. Methods**
- Landmark-based methods
  - [Potamias+, CIKM’09]
- Distance Sketch
  - [Sarma+, WSDM’10]
- Path Sketch
  - [Gubichev+, CIKM’10]
- Landmark-based method
  - + LCA, Shortcuts, ...
    - [Qiao+, ICDE’12]
    - [Tretyakov+, CIKM’12]
Proposed Method: *Pruned Landmark Labeling*

**Exact Methods**
- 2-Hop Cover
  - [Cohen+, SODA’02]
- 2-Hop Cover
  - [Cheng+, EDBT’09]
- Highway 2-Hop
  - [Jin+, SIGMOD’12]
- Hierarchical HL
  - [Abraham+, ESA’12]
- TD-based
  - [Wei, SIGMOD’10]
- TD-based
  - [Akiba+, EDBT’12]

**Approx. Methods**
- Landmark-based methods
  - [Potamias+, CIKM’09]
- Distance Sketch
  - [Sarma+, WSDM’10]
- Path Sketch
  - [Gubichev+, CIKM’10]
- Landmark-based method
  - + LCA, Shortcuts, …
    - [Qiao+, ICDE’12]
    - [Tretyakov+, CIKM’12]

**Pruned Landmark Labeling**
(this work)
Proposed Method: Advantages

Exact Methods

- 2-Hop Cover
  - [Cohen+, SODA'02]
  - [Cheng+, EDBT'09]

Approx. Methods

- Landmark-based methods
  - [Potamias+, CIKM'09]
  - Distance Sketch
    - [Sarma+, WSDM'10]
  - Path Sketch
    - [Gubichev+, CIKM'10]

Pruned Landmark Labeling
- (this work)

Cannot handle networks with >10M edges

Can handle networks with >100M edges

2-Hop Cover
- [Wei, SIGMOD'10]

Hierarchical HL

TD-based

Hierarchical HL

Hierarchical HL

Pruned Landmark Labeling

Can handle networks with >100M edges
Proposed Method: Advantages

Exact Methods

- 2-Hop Cover [Cohen+, SODA’02]
- 2-Hop Cover [Cheng+, EDBT’09]
- Highway 2-Hop [Jin+, SIGMOD’12]
- TD-based [Wei, SIGMOD’10]
- Hierarchical HL

Approx. Methods

- Pruned Landmark Labeling (this work)
- Hierarchical HL [Abraham+, ESA’12]
- 2-Hop Cover [Cheng+, EDBT’09]
- TD-based [Akiba+, EDBT’12]
- Landmark-based methods [Potamias+, CIKM’09]
- Path Sketch [Gubichev+, CIKM’10]
- Landmark-based method [Sarma+, WSDM’10]

No error (exact method)

Bad precision for close pairs
Proposed Method: Advantages

**Exact Methods**

- 2-Hop Cover
  - [Cohen+, SODA'02]
- 2-Hop Cover
  - [Cheng+, EDBT'09]

**Approx. Methods**

- Landmark-based methods
  - [Potamias+, CIKM'09]
- Distance Sketch
  - [Sarma+, WSDM'10]
- Path Sketch
  - [Gubichev+, CIKM'10]
- Landmark-based method
  - + LCA, Shortcuts, ...
  - [Qiao+, ICDE’12]
  - [Tretyakov+, CIKM’12]

**Fast querying (μs)**

- TD-based
  - [Akiba+, EDBT’12]
- Hierarchical HL
  - [Abraham+, ESA’12]
- Pruned Landmark Labeling
  - (this work)

**Slow querying (ms)**
Proposed Method: Advantages

Exact Methods

- 2-Hop Cover
  - [Cohen+, SODA’02]
- Hierarchical HL
  - [Abraham+, ESA’12]
- 2-Hop Cover
  - [Cheng+, EDBT’09]
- Highway 2-Hop
  - [Jin+, SIGMOD’12]
- Landmark-based methods
  - [Potamias+, CIKM’09]

Approx. Methods

- Landmark-based methods
  - [Mueller+, WSDM’10]
- Distance Sketch
  - [Sarma+, EDBT’14]
- Path Sketch
  - [Gubichev+, CIKM’12]

Pruned Landmark Labeling
- (this work)

Simple and easy to implement
Indexing Framework

2-Hop Cover [Cohen+, SODA’02]
Assumption

- Undirected
- Unweighted

(we can easily obtain directed and/or weighted version)
2-Hop Labeling: Index Data Structure

- Commonly used framework (\(= \text{Data Structure} + \text{Query Algo.}\))
  - [Cohen+'02], [Cheng+'09], [Jin+'12], [Abraham+'12] and ours

- Index: label \(L(v) = \{(l_1, \delta_1), (l_2, \delta_2), \ldots\}\)
  - \(l_i \in V, \delta_i = d_G(v, l_i)\)

**Example**

<table>
<thead>
<tr>
<th>Vertex</th>
<th>1</th>
<th>4</th>
<th>5</th>
<th>7</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

\(L(1):\)

<table>
<thead>
<tr>
<th>Vertex</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

\(L(2):\)

<table>
<thead>
<tr>
<th>Vertex</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance</td>
<td>5</td>
<td>0</td>
<td>4</td>
<td>7</td>
<td>2</td>
</tr>
</tbody>
</table>

\(d_G(1,10) = 5\)
2-Hop Labeling: Query Algorithm

- Query: \( d_G(s, t) = \min_{l \in L(s) \cap L(t)} d_G(s, l) + d_G(l, t) \)
  - Paths through common vertices

### Example

<table>
<thead>
<tr>
<th>( L(1) ):</th>
<th>1</th>
<th>4</th>
<th>5</th>
<th>7</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distance</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( L(3) ):</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distance</td>
<td>5</td>
<td>0</td>
<td>4</td>
<td>7</td>
<td>2</td>
</tr>
</tbody>
</table>

Distance between vertex 1 and 3:
- \( 1 \rightarrow \ldots \rightarrow 4 \rightarrow \ldots \rightarrow 3 : 3 + 4 = 7 \) \[ Answer: \min\{6, 7\} = 6 \]
- \( 1 \rightarrow \ldots \rightarrow 7 \rightarrow \ldots \rightarrow 3 : 4 + 2 = 6 \)
2-Hop Labeling: Challenge

Challenge: computing labels

- Correctness (*Exactness*)
- Sizes of labels (*Index Size & Query Time*)
- Efficiency (*Scalability*)

Previous approach [Cohen+’02], [Cheng+’09], [Jin+’12], [Abraham+’12]

- Reduce to optimization problems

Our approach

- Directly assign label entries by graph searches
Pruned Landmark Labeling
Our Approach

1. Naïve Landmark Labeling
   Conduct a BFS from every vertex

2. Pruned Landmark Labeling
   *(proposed method)*
   Pruning during BFSs
Naïve Landmark Labeling (w/o pruning)

1. $L_0 \leftarrow$ an empty index

2. For each vertex $v_1, v_2, \ldots, v_n$
   - Conduct a BFS from $v_i$
   - Label all the visited vertices
     \[ L_i(u) = L_{i-1}(u) \cup (v_i, d_G(u, v_i)) \]

BFS from $v_1$ \hspace{1cm} BFS from $v_2$ \hspace{1cm} BFS from $v_3$ \hspace{1cm} BFS from $v_n$

$L_0 \subseteq L_1 \subseteq L_2 \subseteq \cdots \subseteq (n \text{ times}) \subseteq L_n$

Result

LSNA'14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Naïve Landmark Labeling (w/o pruning)

After a BFS from 1

$L_1(1)$:
- Vertex: 1
- Distance: 0

$L_1(2)$:
- Vertex: 1
- Distance: 2

$L_1(3)$:
- Vertex: 1
- Distance: 2

After a BFS from 2

$L_2(1)$:
- Vertex: 1, 2
- Distance: 0, 2

$L_2(2)$:
- Vertex: 1, 2
- Distance: 2, 0

$L_2(3)$:
- Vertex: 1, 2
- Distance: 2, 3

LSNA'14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Naïve Landmark Labeling (w/o pruning)

1. $L_0 \leftarrow$ an empty index

2. For each vertex $v_1, v_2, \ldots, v_n$
   - Conduct a BFS from $v_i$
   - Label all the visited vertices
     \[ L_i(u) = L_{i-1}(u) \cup (v_i, d_G(u, v_i)) \]

- $\Theta(nm)$ preprocessing time
- $\Theta(n^2)$ space
- Inpractical!
Pruned Landmark Labeling

1. \( L'_0 \leftarrow \text{an empty index} \)

2. For each vertex \( v_1, v_2, ..., v_n \)
   
   - Conduct a pruned BFS from \( v_i \)
   
   - Label all the visited vertices
     
     \[ L'_i(u) = L'_{i-1}(u) \cup (v_i, d_G(u, v_i)) \]

\( \text{PBS from } v_1 \supset L'_0 \supset L'_1 \supset L'_2 \supset \cdots \) (\( n \) times) \supset L'_n \supset \text{Result} \)
If \( \text{QUERY}(v_i, u, L'_{i-1}) \leq \delta \rightarrow \text{Prune } u \)

- We do not label \( u \) this time
- We do not traverse edges from \( u \)
Example

First BFS from vertex 1

\[ L'_1(1): \]
\[ \begin{array}{ll}
   \text{Vertex} & 1 \\
   \text{Distance} & 0 \\
\end{array} \]

\[ L'_1(2): \]
\[ \begin{array}{ll}
   \text{Vertex} & 1 \\
   \text{Distance} & 2 \\
\end{array} \]

\[ L'_1(6): \]
\[ \begin{array}{ll}
   \text{Vertex} & 1 \\
   \text{Distance} & 1 \\
\end{array} \]

Second BFS from vertex 2

\[ \text{QUERY}(2, 6, L'_1) = 2 + 1 = 3 = d(2, 6) \]
\[ \rightarrow \text{Vertex } 6 \text{ is pruned.} \]
Example

The search space gets smaller and smaller
Theorems: Correctness

Theorem 4.1

\[ \text{QUERY}(s, t, L'_i) = \text{QUERY}(s, t, L_i) \]

for any \( s, t \) and \( i \).

Naïve:

\[ L_0 \rightarrow L_1 \rightarrow L_2 \rightarrow \cdots \rightarrow L_n \]

Pruned:

\[ L'_0 \rightarrow L'_1 \rightarrow L'_2 \rightarrow \cdots \rightarrow L'_n \]

"Equal" through function QUERY
Theorems: Correctness

Theorem 4.1

\[ \text{QUERY}(s, t, L'_i) = \text{QUERY}(s, t, L_i) \]

for any \( s, t \) and \( i \).

Corollary 4.1 (Correctness)

\[ \text{QUERY}(s, t, L'_n) = d_G(s, t) \]

for any \( s, t \)

i.e., our method is exact.
**Theorems: Minimality**

**Theorem 4.2 (Minimality)**

$L'_n$ (the constructed index) is minimal.  

_i.e., we cannot remove any label entry from the index._
Vertex Ordering Strategies

Choosing the order of vertices to conduct BFSs from.

• To prune later BFSs as much as possible,
• Central vertices should come first

We conduct BFSs from vertices with higher degree
Pruning on Real-world Networks

Number of vertices labeled in each pruned BFS.

LSNA'14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Pruning on Real-world Networks

Number of vertices labeled in each pruned BFS.

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Theorems: Relations between other methods

Landmark-based approx. methods
[Potamias+’09][Gubichev+’10][Sarma+’10][Qiao+’12][Tretyakov+’12]
Attain high average precision by exploiting hubs

Theorem 4.3
If landmark-based methods attain good precision on a graph
→ then PLL will have small label sizes
PLL can also exploit hubs

After conducting from these hubs, BFSs will be pruned soon
Theorems: Relations between other methods

Tree-decomposition-based methods
[Wei’10], [Akiba+’12]
Attain good performance by exploiting tree-like fringes.
Based on methods for graphs with small tree-width.

Theorem 4.4
PLL perform well on graphs with small tree-width
PLL can also exploit tree-like fringes
Combination of Advantages

**Exact Methods**

- 2-Hop Cover [Cohen+, SODA’02]
- 2-Hop Cover [Cheng+, EDBT’09]
- Highway 2-Hop [Jin+, SIGMOD’12]
- Hierarchical HL [Abraham+, ESA’12]
- TD-based [Wei, SIGMOD’10]
- TD-based [Akiba+, EDBT’12]

**Approx. Methods**

- Landmark-based methods [Potamias+, CIKM’09]
- Distance Sketch [Sarma+, WSDM’10]
- Path Sketch [Gubichev+, CIKM’10]
- Landmark-based method + LCA, Shortcuts, ... [Qiao+, ICDE’12]
  [Tretyakov+, CIKM’12]

**Pruned Landmark Labeling**

(this work)
Combination of Advantages

Exact Methods

- 2-Hop Cover [Cohen+, SODA’02]
- 2-Hop Cover [Cheng+, EDBT’09]
- Highway 2-Hop [Jin+, SIGMOD’12]
- Hierarchical HL [Abraham+, ESA’12]
- TD-based [Akiba+, EDBT’12]
- TD-based [Wei, SIGMOD’10]

Approx. Methods

- Landmark-based methods [Potamias+, CIKM’09]
- Distance Sketch [Sarma+, WSDM’10]
- Path Sketch [Gubichev+, CIKM’10]
- Landmark-based method + LCA, Shortcuts, ... [Qiao+, ICDE’12]
- Existence of highly central nodes

Low tree-width of tree-like fringes

Pruned Landmark Labeling (this work)

LSNA’14, T. Akiba: Pruned Labeling Algorithms: Fast, Exact, Dynamic, Simple and General Indexing Scheme for Shortest-Path Queries
Experiments
Indexing Time

Two orders of magnitude faster

PLL, TD: Xeon X5670, 48GB, Linux, C++, g++, sequential
HHL: Xeon X5680 ×2, 96GB, Windows, C++, VC++, 12 cores
PLL: degree strategy, 16 BP-BFS for first three datasets, 64 BP-BFS for the others
Scalability

Largest networks used in experiments (indexing time: <1day)

Two orders of magnitude larger networks

<table>
<thead>
<tr>
<th>Network</th>
<th>Edges</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEDI [SIGMOD’10]</td>
<td>10^6</td>
<td></td>
</tr>
<tr>
<td>HCL [SIGMOD’12]</td>
<td>10^5</td>
<td></td>
</tr>
<tr>
<td>TD [EDBT’12]</td>
<td>10^4</td>
<td></td>
</tr>
<tr>
<td>HHL [ESA’12]</td>
<td>10^3</td>
<td></td>
</tr>
<tr>
<td>PLL (this work)</td>
<td>10^2</td>
<td></td>
</tr>
</tbody>
</table>
Index Size and Query Time

Index Size

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Index Size (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slashdot (social network)</td>
<td>170</td>
</tr>
<tr>
<td>NotreDame (web graph)</td>
<td>100</td>
</tr>
</tbody>
</table>

Query Time

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Query Time (μs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slashdot (social network)</td>
<td>0.1</td>
</tr>
<tr>
<td>NotreDame (web graph)</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Comparable

PLL, TD: Xeon X5670, 48GB, Linux, C++, g++, sequential
HHL: Xeon X5680 ×2, 96GB, Windows, C++, VC++, 12 cores
PLL: degree strategy, 16 BP-BFS for first three datasets, 64 BP-BFS for the others
Summary of PLL

• Distance queries on Complex Networks

• Proposed method: *Pruned Landmark Labeling*
  – 2-hop cover
  – pruned BFSs + bit-parallel BFSs

• Experiments
  – Scalable
  – Fast
  – (easy to implement)

Software available: [http://git.io/pll](http://git.io/pll)
Extensions of Pruned Landmark Labeling
Reachability Query
[Yano-Akiba-Iwata-Yoshida, CIKM’13]
Given a directed graph $G$, construct an index to answer **reachability queries** asking if there is a path between two vertices.

**Challenge**

Trade-off between **Query Time** and **Scalability**

- Aim to achieve fast query time on large graphs.
Reachability $\subseteq$ Shortest Path

- Reachability queries can be also answered by PLL
- We specialize the method for reachability queries
We first decompose the graph into paths.

PLL: Label for each vertex (=landmark)

PPL: Label for each path
Memorizing “$S$ can reach 2”,
we can easily obtain “$S$ can also reach 3,4,5.”

From “$S \rightarrow 2$” and “$4 \rightarrow T$”, we can obtain “$S \rightarrow T$.”
On 3-hop cover framework [Jin+ ’09] similar to 2-hop cover, we can also construct an index by using pruned BFS

**Idea**

- Decompose $V$ into paths
- Keep reachability to paths
# Results: Average Query Time ($\mu$s)

GRAIL (Yildirim+ ‘12), IL (Schaik+ ’11), PWAH (Schaik+ ’11)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>PLL</th>
<th>PPL</th>
<th>GRAIL</th>
<th>IL</th>
<th>PWAH</th>
</tr>
</thead>
<tbody>
<tr>
<td>ff/successors</td>
<td>0.085</td>
<td>0.133</td>
<td>0.279</td>
<td>0.154</td>
<td>0.202</td>
</tr>
<tr>
<td>uniprot22m</td>
<td>0.083</td>
<td>0.122</td>
<td>0.403</td>
<td>0.173</td>
<td>0.243</td>
</tr>
<tr>
<td>uniprot100m</td>
<td>0.133</td>
<td>0.197</td>
<td>0.743</td>
<td>0.292</td>
<td>0.361</td>
</tr>
<tr>
<td>uniprot150m</td>
<td>0.153</td>
<td>0.223</td>
<td>0.776</td>
<td>0.248</td>
<td>0.351</td>
</tr>
<tr>
<td>citeseerx</td>
<td>0.124</td>
<td>0.164</td>
<td>27.946</td>
<td>0.103</td>
<td>0.214</td>
</tr>
<tr>
<td>cit-patents</td>
<td>0.253</td>
<td>0.296</td>
<td>11.591</td>
<td>0.292</td>
<td>15.451</td>
</tr>
<tr>
<td>go-uniprot</td>
<td>0.156</td>
<td>0.194</td>
<td>0.520</td>
<td>0.233</td>
<td>0.521</td>
</tr>
</tbody>
</table>

Graphs with 10M-edges:

- 1st and 2nd in query time on almost all datasets
- Very slow on some datasets
# Results: Index Size (MB)

## Better than PLL on one dataset

<table>
<thead>
<tr>
<th>Dataset</th>
<th>PLL</th>
<th>PPL</th>
<th>GRAIL</th>
<th>IL</th>
<th>PWAH</th>
</tr>
</thead>
<tbody>
<tr>
<td>ff/successors</td>
<td>122.3</td>
<td>91.6</td>
<td>29.7</td>
<td>40.0</td>
<td>34.1</td>
</tr>
<tr>
<td>uniprot22m</td>
<td>19.4</td>
<td>19.4</td>
<td>25.5</td>
<td>19.6</td>
<td>19.5</td>
</tr>
<tr>
<td>uniprot100m</td>
<td>206.8</td>
<td>206.8</td>
<td>257.4</td>
<td>223.0</td>
<td>218.8</td>
</tr>
<tr>
<td>uniprot150m</td>
<td>334.0</td>
<td>334.0</td>
<td>400.6</td>
<td>373.8</td>
<td>366.2</td>
</tr>
<tr>
<td>citeseerx</td>
<td>122.0</td>
<td>126.7</td>
<td>104.6</td>
<td>441.3</td>
<td>156.0</td>
</tr>
<tr>
<td>cit-patents</td>
<td>664.6</td>
<td>691.2</td>
<td>60.4</td>
<td>22444</td>
<td>5593</td>
</tr>
<tr>
<td>go-uniprot</td>
<td>263.1</td>
<td>273.5</td>
<td>111.5</td>
<td>792.7</td>
<td>255.9</td>
</tr>
</tbody>
</table>

**Moderate index size**

**Index size exploded on one large dataset**
Distance Query on Road Networks
[Akiba-Iwata-Kawata-Kawarabayashi, ALENEX’14]
Shortest-path Queries

Road Network

Preprocess

Index

Query\((s, t)\)

Distance\(d(s, t)\)

Goal

- Fast preprocessing time
- Small index size
- Fast query time
Structural Difference

- Complex networks
  - Exceptionally central hubs (=landmarks)

- Road networks
  - No such very central hubs
  - But there are central paths (=highways)
Labeling for Paths

Can cover all shortest paths that hit the path

Use highways efficiently
Labeling for Paths

Similarity with distance oracles for planar graphs [Thorup,04]

Separate the graph by paths
Label $L(v)$ is a set of triples $(i, d(p_{i,1}, p_{i,j}), d(v, p_{i,j}))$.

Distance on the path

$$
Query(s, t, L) = \min \{d(s, p_{i,j}) + d(p_{i,j}, p_{i,k}) + d(p_{i,k}, t) | \\
(i, d(p_{i,1}, p_{i,j}), d(s, p_{i,j})) \in L(s), \\
(i, d(p_{i,1}, p_{i,k}), d(t, p_{i,k})) \in L(t)\}
$$
## Experimental Results

| USA (|V|=24M, |E|=58M) | Preprocessing | Space | Query |
|----------------|---------------|--------|-------|
| **CH** [Geisberger+,08] | 0:27 | 0.5 | 130000 |
| **TNR** [Bast+,07] | 1:30 | 5.4 | 3000 |
| **TNR+AF** [Bauer+,10] | 2:37 | 6.3 | 1700 |
| **HL local** [Abraham+,11] | 2:24(×12) *2 | 22.7 | 627 |
| **HL global** [Abraham+,11] | 2:35(×12) *2 | 25.4 | 266 |
| **HL-15 local** [Abraham+,12] | - | - | - |
| **HL-∞ global** [Abraham+,12] | - | - | - |
| **HLC-15** [Delling+,13] | 0:53 | 2.9 | 2486 |
| **PHL-1** | 0:29 | 16.4 | 941 |

*1 CH,TNR,TNR+AF: AMD Opteron 270 (2.0 GHz) HL,HLC: Intel Xeon X5680 (3.33 GHz) PHL: Intel Xeon X5670 (3.06 GHz)  
*2 parallelized to use 12 cores
Dynamic and Historical
[Akiba-Iwata-Yoshida, WWW’14]
Contributions

Pruned Landmark Labeling [SIGMOD’13]
State-of-the-art static exact method

Contribution 1

Dynamic Pruned Landmark Labeling

Contribution 2

Historical Pruned Landmark Labeling
Contribution 1: +Dynamic Update

Dynamic Pruned Landmark Labeling

Indexing

Incremental Update

*In milliseconds*
**Contribution 2: +Historical Queries**

**Historical Pruned Landmark Labeling**

Time

Query: transition of distance

Distance

Time

Index
For Details...

Please come along to my talk tomorrow 😊

**Session**: Social Networks 1
April 9 (Wed.) 16:00-17:30 @ Room 101
Conclusions

Pruned Labeling Algorithms

• Unified approach, simple but effective

Queries

• Distance queries on complex networks
• Distance queries on road networks
• Reachability queries
• Dynamic update and historical queries