Querying Shortest Distance on Large Graphs

Miao Qiao, Hong Cheng, Lijun Chang and Jeffrey Xu Yu

Department of Systems Engineering & Engineering Management
The Chinese University of Hong Kong

October 19, 2011

Roadmap

- Preliminary
- Related Work
- Problem Statement
- Query-Dependent Local Landmark Scheme
- Optimization Techniques
- Experiments
- Conclusion

Landmark Embedding

Consider $S = \{l_1, \ldots, l_k\} \subseteq V$ which are called *landmarks*. For each $l_i \in S$, we compute the shortest distances to all nodes in V. Then for $\forall v \in V$, it has a k-dimensional vector representation:

$$\overrightarrow{D}(v) = \langle \delta(l_1, v), \delta(l_2, v), \ldots, \delta(l_k, v) \rangle$$

Landmark Embedding

Consider $S = \{l_1, \ldots, l_k\} \subseteq V$ which are called *landmarks*. For each $l_i \in S$, we compute the shortest distances to all nodes in V. Then for $\forall v \in V$, it has a k-dimensional vector representation:

$$\overrightarrow{D}(v) = \langle \delta(l_1, v), \delta(l_2, v), \ldots, \delta(l_k, v) \rangle$$

Given $q = (a, b)$, we estimate shortest distance with *landmark embedding* based on *triangle inequality* as

$$\widetilde{\delta}(a, b) = \min_{l_i \in S} \{\delta(l_i, a) + \delta(l_i, b)\}$$

Landmark Selection Strategy

Selecting the optimal set of landmarks is very hard!

- Betweenness centrality based criterion, NP-hard (Potamias et al. [8])
- Minimum K-center formulation, NP-hard (Francis et al. [2])

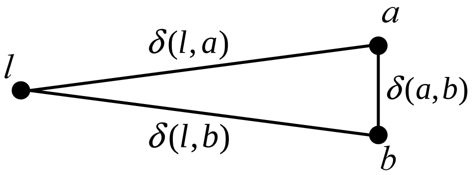
Graph measure based heuristics:

- Random selection
- Degree based landmark selection [2]
- Approximate centrality based landmark selection [8]

Performance largely depends on graph properties, e.g., degree distribution, diameter, etc.

Embedding Performance

- A query-independent global landmark set S is not accurate in estimating shortest distances.



$$\tilde{\delta}(a,b) = \delta(l,a) + \delta(l,b) >> \delta(a,b)$$

Embedding Performance

Increasing the number of landmarks k improves the estimation accuracy, but also increases the complexity.

- Query time $O(k)$
- Index time $O(kn \log n)$
- Index size $O(kn)$

Here $k = |S|$ and $n = |V|$.

Roadmap

- Preliminary
- Related Work
- Problem Statement
- Query-Dependent Local Landmark Scheme
- Optimization Techniques
- Experiments
- Conclusion

Related Work

Landmark embedding has been widely used in estimating shortest distances in

- Road networks [13, 5]
- Social networks and web graphs [9, 8, 12, 3]
- Internet [2, 6]

Related Work

Landmark embedding has been widely used in estimating shortest distances in

- Road networks [13, 5]
- Social networks and web graphs [9, 8, 12, 3]
- Internet [2, 6]

Major differences between these methods lie in three aspects:

- Landmark selection: random, degree, betweenness centrality, coverage, etc.
- Landmark organization: flat or hierarchical organization
- Error bound or not: Thorup and Zwick [14] provide $(2k-1)$-approximation with $O(kn^{1+1/k})$ memory

Sketch Based Distance oracle [12, 3]

Let $t = \lfloor \log n \rfloor$ where $n = |V|$. Uniformly sample $t + 1$ sets of landmarks (also called seed sets) of sizes $2^0, 2^1, 2^2, \ldots, 2^t$.

Figure: Multiple Seed Sets

Sketch Based Distance oracle [12, 3]

Let $t = \lfloor \log n \rfloor$ where $n = |V|$. Uniformly sample $t + 1$ sets of landmarks (also called seed sets) of sizes $2^0, 2^1, 2^2, \ldots, 2^t$.

Layer 1 — 2^0
Layer 2 — 2^1

Figure: Multiple Seed Sets

Sketch Based Distance oracle [12, 3]

Let $t = \lfloor \log n \rfloor$ where $n = |V|$. Uniformly sample $t + 1$ sets of landmarks (also called seed sets) of sizes $2^0, 2^1, 2^2, \ldots, 2^t$.

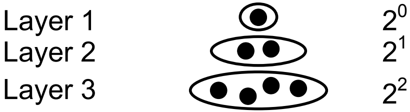


Figure: Multiple Seed Sets

Sketch Based Distance oracle [12, 3]

Let $t = \lfloor \log n \rfloor$ where $n = |V|$. Uniformly sample $t + 1$ sets of landmarks (also called seed sets) of sizes $2^0, 2^1, 2^2, \ldots, 2^t$.

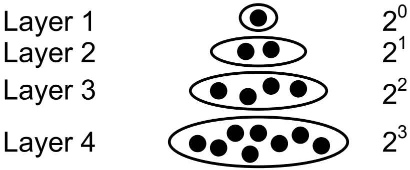


Figure: Multiple Seed Sets

Sketch Based Distance oracle [12, 3]

Let $t = \lfloor \log n \rfloor$ where $n = |V|$. Uniformly sample $t + 1$ sets of landmarks (also called seed sets) of sizes $2^0, 2^1, 2^2, \ldots, 2^t$.

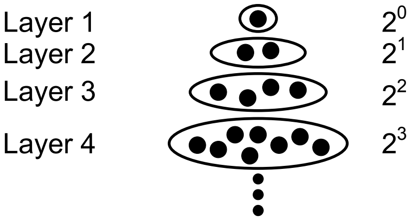


Figure: Multiple Seed Sets

Sketch Based Distance oracle [12, 3]

Let $t = \lfloor \log n \rfloor$ where $n = |V|$. Uniformly sample $t + 1$ sets of landmarks (also called seed sets) of sizes $2^0, 2^1, 2^2, \ldots, 2^t$.

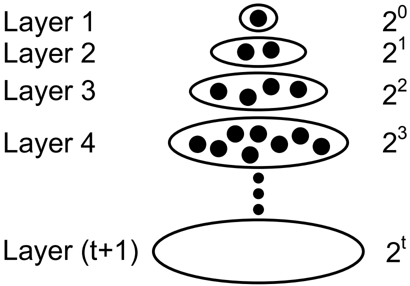


Figure: Multiple Seed Sets

Sketch Based Distance oracle [12, 3]

Let $t = \lfloor \log n \rfloor$ where $n = |V|$. Uniformly sample $t + 1$ sets of landmarks (also called seed sets) of sizes $2^0, 2^1, 2^2, \ldots, 2^t$.

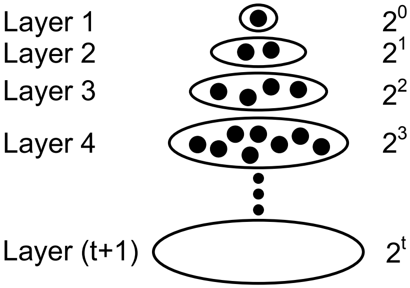


Figure: Multiple Seed Sets

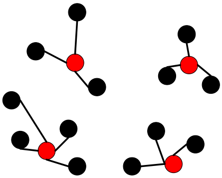


Figure: Index with Layer 2 Landmarks

K-Nearest Neighbor and Shortest Path Query on Spatial Networks

- Euclidean distance as a lower bound to prune the search space (Papadias et al. [7])
- First order Voronoi diagram for KNN query (Kolahdouzan and Shahabi [4])
- Shortest path quadtree for KNN query (Samet et al. [10])
- Path-distance oracle of size $O(n \cdot \max(s^d, \frac{1}{\epsilon}^d))$, and answer a query with ϵ-approximation in $O(\log n)$ time (Sankaranarayanan en al. [11])
- etc.

Roadmap

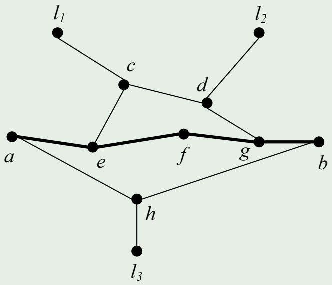
- Preliminary
- Related Work
- Problem Statement
- Query-Dependent Local Landmark Scheme
- Optimization Techniques
- Experiments
- Conclusion

Example

Landmark set
$S = \{l_1, l_2, l_3\}$.

For a query (a, b),
$P(a, b) = (a, e, f, g, b)$.

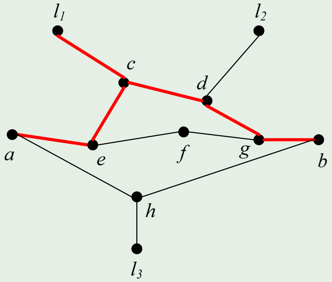
$P(l_1, a) = (l_1, c, e, a)$,
$P(l_1, b) = (l_1, c, d, g, b)$



Example

Landmark set
$S = \{l_1, l_2, l_3\}$.

For a query (a, b),
$P(a, b) = (a, e, f, g, b)$.



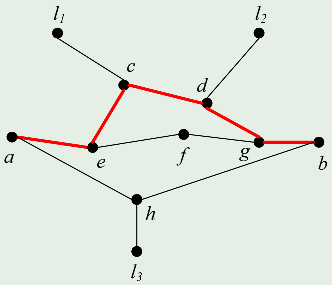
$P(l_1, a) = (l_1, c, e, a)$,
$P(l_1, b) = (l_1, c, d, g, b)$
Based on landmark l_1,

$$\widetilde{\delta}(a, b) = \delta(l_1, a) + \delta(l_1, b)$$

Example

Landmark set
$S = \{l_1, l_2, l_3\}$.

For a query (a, b),
$P(a, b) = (a, e, f, g, b)$.



$P(l_1, a) = (l_1, c, e, a)$,
$P(l_1, b) = (l_1, c, d, g, b)$
Based on landmark l_1,

$$\widetilde{\delta}(a, b) = \delta(l_1, a) + \delta(l_1, b)$$

But based on node c,

$$\widetilde{\delta}^L(a, b) = \delta(c, a) + \delta(c, b)$$

is more accurate because

$$\widetilde{\delta}(a, b) = \widetilde{\delta}^L(a, b) + 2\delta(l_1, c)$$

.

Query-Dependent Local Landmarks

Given a global landmark set S and a query (a, b), a query-dependent local landmark function is

$$L_{ab}(S) : V^k \mapsto V$$

which maps S to a vertex in V, called a local landmark, depending on the query nodes a and b.

Query-Dependent Local Landmarks

Given a global landmark set S and a query (a, b), a query-dependent local landmark function is

$$L_{ab}(S) : V^k \mapsto V$$

which maps S to a vertex in V, called a local landmark, depending on the query nodes a and b.

With the local landmark function, we can estimate a shortest distance of a query (a, b) as

$$\widetilde{\delta}^L(a, b) = \delta(L_{ab}(S), a) + \delta(L_{ab}(S), b)$$

which can provide a more accurate distance estimation.

Roadmap

- Preliminary
- Related Work
- Problem Statement
- Query-Dependent Local Landmark Scheme
- Optimization Techniques
- Experiments
- Conclusion

Shortest Path Tree

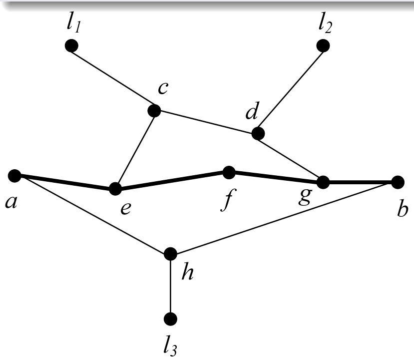
Definition (Shortest Path Tree)

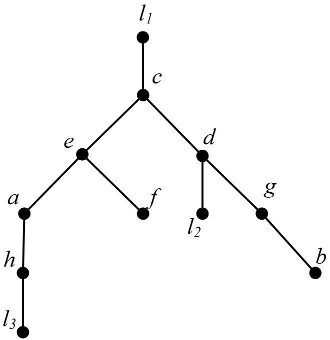
Given a graph $G = (V, E)$, the shortest path tree rooted at a vertex $r \in V$ is a spanning tree of G, such that the path from the root r to each node $v \in V$ is a shortest path between r and v.

Shortest Path Tree

Definition (Shortest Path Tree)

Given a graph $G = (V, E)$, the shortest path tree rooted at a vertex $r \in V$ is a spanning tree of G, such that the path from the root r to each node $v \in V$ is a shortest path between r and v.





SPT Based Local Landmark Function

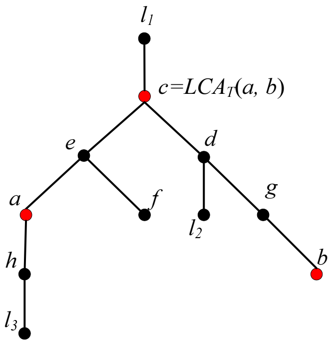
Definition (SPT Based Local Landmark Function)

Given a global landmark set S and a query (a, b), the SPT based local landmark function is defined as:

$$L_{ab}(S) = \arg \min_{r \in \{LCA_{T_l}(a,b) | l \in S\}} \{\delta(r, a) + \delta(r, b)\}$$

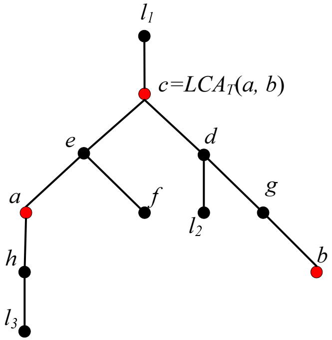
where $LCA_{T_l}(a, b)$ denotes the least common ancestor of a and b in the shortest path tree T_l rooted at $l \in S$.

Least Common Ancestor



$$\widetilde{\delta}^{L}(a,b) = \delta(c,a) + \delta(c,b)$$
$$= \delta(l_1,a) + \delta(l_1,b) - 2\delta(l_1,c)$$

Least Common Ancestor



$$\widetilde{\delta}^L(a,b) = \delta(c,a) + \delta(c,b)$$
$$= \delta(l_1,a) + \delta(l_1,b) - 2\delta(l_1,c)$$

Theorem

Given a global landmark set S, $\forall a, b \in V$, we have

$$\delta(a,b) \leq \widetilde{\delta}^L(a,b) \leq \widetilde{\delta}(a,b)$$

Efficient LCA computation - Transform LCA to RMQ

Definition (Range Minimum Query)

Let $A[1, \ldots, n]$ be an array. For indices $1 \leq i \leq j \leq n$,

$$RMQ_A(i, j) = \arg \min_{i \leq k \leq j} \{A[k]\}$$

Efficient LCA computation - Transform LCA to RMQ

Definition (Range Minimum Query)

Let $A[1, \ldots, n]$ be an array. For indices $1 \leq i \leq j \leq n$,

$$RMQ_A(i, j) = \arg \min_{i \leq k \leq j} \{A[k]\}$$

Observation

$LCA(a, b)$ is the shallowest node encountered between the visits to a and to b during a DFS of a tree T.

Efficient LCA computation - Transform LCA to RMQ

Transform LCA to RMQ:

- Perform a DFS search on an SPT T and record the sequence of nodes visited
- Record the level of each node in the tree as its distance to the root
- For two query nodes a, b, use an RMQ query to find the node between them with the smallest level
- Such a node is $LCA_T(a, b)$

Efficient LCA computation - Transform LCA to RMQ

Transform LCA to RMQ:

- Perform a DFS search on an SPT T and record the sequence of nodes visited
- Record the level of each node in the tree as its distance to the root
- For two query nodes a, b, use an RMQ query to find the node between them with the smallest level
- Such a node is $LCA_T(a, b)$

The complexities of the state-of-the-art RMQ technique [1] are:

- Index Time $O(n)$
- Index Size $O(n)$
- Query Time $O(1)$

Complexity of The Local Landmark Embedding Scheme

- Online Query Time Complexity: $O(k)$
- Offline Embedding Space Complexity: $O(kn)$
- Offline Embedding Time Complexity: $O(kn \log n)$

where $k = |S|$ and $n = |V|$.

The query-dependent local landmark embedding has the SAME complexities as the global landmark embedding.

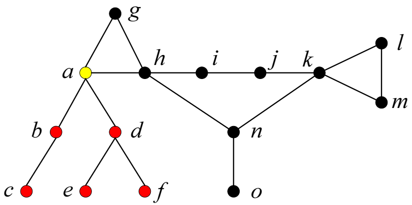
Roadmap

- Preliminary
- Related Work
- Problem Statement
- Query-Dependent Local Landmark Scheme
- Optimization Techniques
  - Index Reduction with Graph Compression
  - Improving the Accuracy by Local Search
- Experiments
- Conclusion

Index Reduction with Graph Compression

A graph can be compressed by removing two special types of nodes:

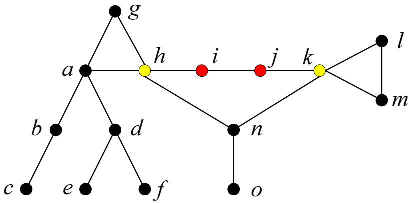
- Tree node
  - Map to the root node and record its distance to the root
  - Remove the tree node



Index Reduction with Graph Compression

A graph can be compressed by removing two special types of nodes:

- Tree node
  - Map to the root node and record its distance to the root
  - Remove the tree node
- Chain node
  - Map to two end nodes and record its distances to both ends
  - Remove the chain node



Index Reduction with Graph Compression

Query time: $O(1)$

$$\widetilde{\delta}^L(a, b) = \min_{r_a \in map(a), r_b \in map(b)} \{\delta(a, r_a) + \widetilde{\delta}^L(r_a, r_b) + \delta(b, r_b)\}$$

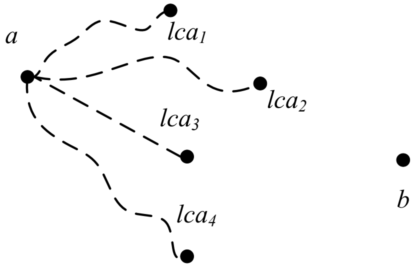
where $map(a)$ contains the nodes that a maps to, i.e.,

- $map(a)$ contains a root node, if a is a tree node
- $map(a)$ contains two end nodes, if a is a chain node
- $map(a)$ contains a itself, otherwise

Index size: reduce size to $O(n + (|S| - 1)n')$ from $O(|S|n)$, where n' and n are the number of nodes in the compressed graph and in the original graph, respectively.

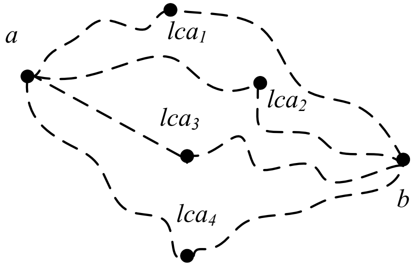
Improving the Accuracy by Local Search

- Connect two query nodes to all local landmarks through the shortest paths



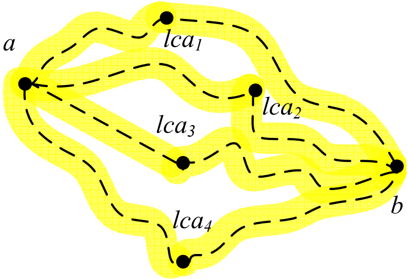
Improving the Accuracy by Local Search

- Connect two query nodes to all local landmarks through the shortest paths



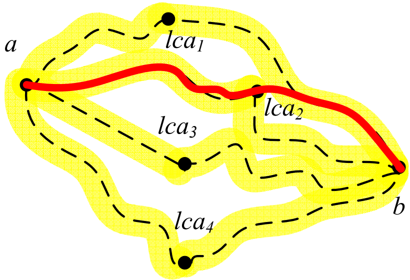
Improving the Accuracy by Local Search

- Connect two query nodes to all local landmarks through the shortest paths
- Expand each node to include its c-hop neighbors



Improving the Accuracy by Local Search

- Connect two query nodes to all local landmarks through the shortest paths
- Expand each node to include its *c*-hop neighbors
- The expanded nodes may form shortcuts which provide tighter distance estimation



Roadmap

- Preliminary
- Related work
- Problem Statement
- Query-Dependent Local Landmark Scheme
- Optimization Techniques
- Experiments
- Conclusion

Dataset Description

Table: Network Statistics

Dataset	$\|V\|$	$\|E\|$	$\|V'\|$	$\|E'\|$
Slashdot	77,360	905,468	36,012	752,478
Google	875,713	5,105,039	449,341	4,621,002
Youtube	1,157,827	4,945,382	313,479	4,082,046
Flickr	1,846,198	22,613,981	493,525	18,470,294
NYRN	264,346	733,846	164,843	532,264
USARN	23,947,347	58,333,344	7,911,536	24,882,476

$|V'|$ and $|E'|$ denote the number of nodes and edges in the compressed graph.

Comparison Methods and Metrics

The embedding methods for comparison:

- Global Landmark Scheme (GLS)
- Local Landmark Scheme (LLS)
- Local Search (LS)
- 2RNE [5]
- TreeSketch [3]

Comparison Methods and Metrics

The embedding methods for comparison:

- Global Landmark Scheme (GLS)
- Local Landmark Scheme (LLS)
- Local Search (LS)
- 2RNE [5]
- TreeSketch [3]

Evaluation Metrics: relative error

$$err = \frac{|\widetilde{\delta}(s,t) - \delta(s,t)|}{\delta(s,t)}$$

Average Relative Error

		SlashD	Google	Youtube	Flickr	NYRN	USARN
				$k = 20$			
Rand	GLS	0.6309	0.5072	0.6346	0.5131	0.1825	0.1121
	LLS	0.1423	0.0321	0.0637	0.0814	0.0246	0.0786
	LS	0.0000	0.0046	0.0009	0.0001	0.0071	0.0090
Cent	GLS	0.1520	0.0426	0.0595	0.0567	0.6458	1.5599
	LLS	0.1043	0.0290	0.0489	0.0503	0.1536	0.4708
	LS	0.0001	0.0074	0.0010	0.0003	0.1479	0.4703
				$k = 50$			
Rand	GLS	0.4535	0.4750	0.4549	0.4559	0.1188	0.0632
	LLS	0.0727	0.0142	0.0391	0.0444	0.0103	0.0241
	LS	0.0000	0.0022	0.0003	0.0001	0.0042	0.0030
Cent	GLS	0.1385	0.0245	0.0461	0.0524	0.6133	0.7422
	LLS	0.0663	0.0140	0.0334	0.0284	0.1533	0.4505
	LS	0.0000	0.0037	0.0005	0.0000	0.1455	0.4483

Online Query Time in Milliseconds

	SlashD	Google	Youtube	Flickr	NYRN	USARN
			$k = 20$			
GLS	0.002	0.005	0.008	0.009	0.006	0.020
LLS	0.006	0.021	0.015	0.014	0.036	0.067
LS	0.158	2.729	2.818	4.735	0.681	58.289
			$k = 50$			
GLS	0.005	0.016	0.024	0.027	0.014	0.058
LLS	0.018	0.054	0.032	0.033	0.091	0.196
LS	0.527	3.492	4.178	6.817	1.585	98.221

Index Size in MB

	SlashD	Google	Youtube	Flickr	NYRN	USARN
			$k = 20$			
GLS	6.2	57.9	90.7	124.7	21.2	1915.8
LLS	10.4	122.7	103.2	156.1	85.3	4424.6
LS	16.4	159.7	135.9	303.9	89.6	4623.6
			$k = 50$			
GLS	15.5	144.8	226.8	311.6	52.9	4789.5
LLS	23.3	284.5	216.1	333.8	203.9	9948.3
LS	29.4	321.4	248.7	481.6	208.2	10147.3

Comparison with Other Methods

Dataset	Algorithm	AvgErr	Query Time(ms)	Index Size(MB)
SlashD	2RNE	0.8345	0.001	6.2
	TreeSketch	0.0011	0.176	37.4
	LS	0.0000	0.158	16.4
Google	2RNE	0.5750	0.001	57.9
	TreeSketch	0.0048	3.549	383.7
	LS	0.0046	2.729	159.7
Youtube	2RNE	0.7138	0.001	90.7
	TreeSketch	0.0005	5.317	587.7
	LS	0.0009	2.818	135.9
Flickr	2RNE	0.6233	0.001	124.7
	TreeSketch	0.0001	7.333	959.6
	LS	0.0001	4.735	303.9
NYRN	2RNE	0.4748	0.001	21.2
	TreeSketch	0.0156	1.074	120.5
	LS	0.0071	0.681	89.6
USARN	2RNE	0.4240	0.002	1915.8
	TreeSketch	0.0379	104.769	14555.5
	LS	0.0090	58.289	4623.6

Roadmap

- Preliminary
- Related work
- Problem Statement
- Query-Dependent Local Landmark Scheme
- Optimization Techniques
- Experiments
- Conclusion

Conclusion

- We proposed a query-dependent local landmark scheme, which is more accurate than GLS and has the same complexities.

Conclusion

- We proposed a query-dependent local landmark scheme, which is more accurate than GLS and has the same complexities.
- The local landmark scheme provides very accurate distance estimation, with little dependency on the global landmark selection strategy or the global landmark number.

Conclusion

- We proposed a query-dependent local landmark scheme, which is more accurate than GLS and has the same complexities.
- The local landmark scheme provides very accurate distance estimation, with little dependency on the global landmark selection strategy or the global landmark number.
- The local landmark is computed at query time with an $O(1)$ RMQ operation. This is different from the sketch based methods [12, 3], which build multiple landmark sets apriori to reduce the estimation error.

Q&A

Thanks!

M. A. Bender and M. Farach-Colton.
The LCA problem revisited.
In *LATIN 2000: Theoretical Informatics*, volume 1776 of *Lecture Notes in Computer Science*, pages 88–94. Springer Berlin / Heidelberg, 2000.

P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang.
IDMaps: A global internet host distance estimation service.
IEEE/ACM Trans. Networking, 9(5):525–540, 2001.

A. Gubichev, S. Bedathur, S. Seufert, and G. Weikum.
Fast and accurate estimation of shortest paths in large graphs.
In *Proc. Int. Conf. Information and Knowledge Management (CIKM'10)*, 2010.

M. Kolahdouzan and C. Shahabi.
Voronoi-based k nearest neighbor search for spatial network databases.

In *Proc. Int. Conf. Very Large Data Bases (VLDB'04)*, pages 840–851, 2004.

H.-P. Kriegel, P. Kröger, M. Renz, and T. Schmidt.
Hierarchical graph embedding for efficient query processing in very large traffic networks.
In *Proc. Int. Conf. Scientific and Statistical Database Management (SSDBM'08)*, pages 150–167, 2008.

T. S. E. Ng and H. Zhang.
Predicting internet network distance with coordinates-based approaches.
In *Int. Conf. on Computer Communications (INFOCOM'01)*, pages 170–179, 2001.

D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao.
Query processing in spatial network database.
In *Proc. Int. Conf. Very Large Data Bases (VLDB'03)*, pages 802–813, 2003.

M. Potamias, F. Bonchi, C. Castillo, and A. Gionis.
Fast shortest path distance estimation in large networks.
In *Proc. Int. Conf. Information and Knowledge Management (CIKM'09)*, pages 867–876, 2009.

M. J. Rattigan, M. Maier, and D. Jensen.
Using structure indices for efficient approximation of network properties.
In *Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'06)*, pages 357–366, 2006.

H. Samet, J. Sankaranarayanan, and H. Alborzi.
Scalable network distance browsing in spatial databases.
In *Proc. ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'08)*, pages 43–54, 2008.

J. Sankaranarayanan, H. Samet, and H. Alborzi.
Path oracles for spatial networks.
PVLDB, pages 1210–1221, 2009.

A. D. Sarma, S. Gollapudi, M. Najork, and R. Panigrahy.
A sketch-based distance oracle for web-scale graphs.
In *Proc. Int. Conf. Web Search and Data Mining (WSDM'10)*, pages 401–410, 2010.

C. Shahabi, M. Kolahdouzan, and M. Sharifzadeh.
A road network embedding technique for k-nearest neighbor search in moving object databases.
In *Proc. 10th ACM Int. Symp. Advances in Geographic Information Systems (GIS'02)*, pages 94–100, 2002.

M. Thorup and U. Zwick.
Approximate distance oracles.
Journal of the ACM, 52(1):1–24, 2005.