Adaptive Landmark Selection Strategies for Fast Shortest Path Computation in Large Real-World Graphs

Frank Takes & Walter Kosters

Leiden University, The Netherlands

Web Intelligence 2014 — August 11–14, 2014
Overview

- Introduction
- Distance Queries
- Landmark Framework
- Adaptive Landmark Selection
- Experiments & Results
- Conclusion
Figure: Social network with 1422 nodes and 3711 undirected edges. Node size proportional to degree, color proportional to node PageRank.
Figure: PPI network with 1,458 nodes and 1,948 undirected edges. Node size proportional to degree, color proportional to node eccentricity.
Graphs . . .

- Graphs are **everywhere**: (online) social networks, collaboration networks, webgraphs, communication networks, etc.
- Real-world graphs can be **large**: a graph with 8 million nodes and 1 billion links is not uncommon.
- Graphs have all kinds of interesting **structural properties**: distance distribution, radius, diameter, etc.
  - Common underlying **computational task**: find a shortest path (compute the distance) from $A$ to $B$. 

Figure: Distance distribution of an online social network with 8 million nodes and 1 billion links, sampled over 100,000 node pairs.
Distance Queries

- Unweighted Graph $G = (V, E)$ with:
  - $|V| = n$ nodes
  - $|E| = m$ links/edges

- $N(v)$ and $N'(v)$ are used to denote sets of nodes with a link respectively from and to a node $v \in V$

- **Distance query**: Given nodes $u, v \in V$, compute $d(u, v)$

- Weighted graphs: Dijkstra’s Algorithm

- Unweighted graphs: Breadth First Search (BFS)
Breadth First Search (BFS)

- From the starting node, visit neighboring nodes in level-order until the goal node is found.
- One BFS takes $O(m)$ time — 6 seconds if $m = 1$ billion.
- One BFS for each of the $n$ nodes = All Pairs Shortest Path (APSP) with time complexity $O(mn)$.
- **Research question**: How to accurately answer thousands of distance queries in 1 second? Some (short) precomputation time will be given.
“Rules of the Game”

- **Precomputation phase**: should not be more complex than a few hundred BFSes, in a large graph no more than $c \cdot m$ for a small integer $c < \log(mn)$.

- **Query phase**: answering distance queries using the precomputed values.

- **Efficiency**: query time should be very short, say $O(n)$ or linear in some integer smaller than $n$.

- **Accuracy**: distance estimation should be accurate: a low error according to some error measure.
Landmarks

Precomputation phase

- Landmark set $L \subseteq V$ with $|L| \ll |V|$
- Common landmark count: $|L| = 100$
- Precompute for all $u \in L$ and $v \in V$ the value of $d(u, v)$
  So, perform 100 full BFSes and store the result.

Query phase: for distance query $d(v, w)$, with $v, w \in V$, return:

- 0 if $v = w$ \(O(1)\)
- 1 if $w \in N(v)$ \(O(\log(m/n))\)
- 2 if $N(v) \cap N'(w) \neq \emptyset$ \(O(m/n)\)
- $\min_{u \in L} (d(v, u) + d(u, w))$ otherwise \([1]\) \(O(|L|)\)
Optimizations

- Shortcuts: $(A, B, C, D) \rightarrow (A, B, D)$ if $(B, D) \in E$ [2]
- Cycle elimination: $(V, X, Y, X, Z) \rightarrow (V, X, Z)$ [2]
- Degree-1 checks: if $\text{deg}(u) = 1$ and $(u, v) \in E$ then for all $w \in V$ it holds that $d(u, w) = d(v, w) + 1$.
- Graph homomorphism: if one node maps to another node according to some homomorphism function $h$, then their distance to nodes for which $h(v) = v$, are equal.
Landmark Example (0)

**Landmarks $F$ and $P$**

Precompute for all $v \in V$: $d(F, v)$ and $d(P, v)$

![Diagram showing a network with landmarks $F$ and $P$]
Compute $d(L, J)$

$L \in N(J)$, so $d(J, L) = 1$
Landmark Example (2)

**Compute** $d(B, D)$

$N(B) \cap N(D) \neq \emptyset$, so $d(B, D) = 2$
Landmark Example (3)

Compute $d(F, Q)$
via $F$: $d(F, F) + d(F, Q) = 0 + 4 = 4$
Landmark Example (4)

Compute $d(E, N)$
via $P$: $d(E, P) + d(P, N) = 4 + 1 = 5$
via $F$: $d(E, F) + d(F, N) = 1 + 3 = 4$
Landmark Example (5)

**Compute** $d(G, M)$

via $P$: $(G, J, L, P, L, M)$ with length 5
via $F$: $(G, F, J, L, M)$ with length 4, but $(J, G) \in E$
so eliminating $F$ gives $(G, J, L, M)$ with length 3.
Landmark Framework (1)

- Landmark selection
  - Deciding if some set of landmarks is optimal, is NP-hard
  - Baseline: a random landmark set $L$ from the node set $V$ [3]
  - Better: select the top-$|L|$ nodes from the node list, sorted using some centrality measure:
    - Degree centrality
    - PageRank centrality
    - Betweenness centrality
- Landmark processing: process the list of candidate landmarks in a “smart way”
  - Skip direct neighbors
Selection using Centrality Measures

Figure: Performance of different centrality measures for selecting landmarks on the 21K node CA-CONDMAT network.
Adaptive landmark selection

1. Sort the set of nodes using degree centrality
2. Perform a number of sample BFS runs and store how many times a node \( v \) occurs on a shortest path
3. Compute the increase of success rate of each node in the sorted list compared to the previous node
4. Re-sort the list of nodes according to the contribution to the success rate
5. Take the top-\( k \) nodes from this list

Greedy central neighbor (GCN) processing: select \( h > 0 \) times a selected node’s most central neighbor (according to some centrality measure), if such a neighbor exists
Comparison of Landmark Strategies

**CA-HepPH network:**
- Scientific collaboration network, field of high energy physics
- \( n = 11\,200 \) nodes
- \( m = 235\,000 \) edges
- Average node-node distance of 4.66

Compute for each landmark selection strategy the **node error** as follows:

\[
\frac{|d_{\text{real}} - d_{\text{estimate}}|}{d_{\text{real}}}
\]
Node Errors — Degree Gentrality
Node Errors — GCN-processing
### Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Type</th>
<th>$n$</th>
<th>$m$</th>
<th>$\bar{d}$</th>
<th>Random 0 gcn ($h$)</th>
<th>Random 0 skip-1 gcn ($h$)</th>
<th>Betweenness 0 skip-1 gcn ($h$)</th>
<th>PageRank 0 skip-1 gcn ($h$)</th>
<th>Degree 0 skip-1 gcn ($h$)</th>
<th>Adaptive 0 skip-1 gcn ($h$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA-HepPh</td>
<td>collaboration</td>
<td>11.2K</td>
<td>235K</td>
<td>4.66</td>
<td>.059</td>
<td>.080 (3)</td>
<td>.045 .078 .045 (2)</td>
<td>.140 .090 .093 (3)</td>
<td>.137 .091</td>
<td>.117 .103 .080 (3)</td>
</tr>
<tr>
<td>GoogleNw</td>
<td>web</td>
<td>15.7K</td>
<td>297K</td>
<td>2.46</td>
<td>.224 .000 (3)</td>
<td>.001 .004 .000 (3)</td>
<td>.001 .003 .001 (2)</td>
<td>.001 .003</td>
<td>.001 .003</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>CA-CondMat</td>
<td>collaboration</td>
<td>21.3K</td>
<td>182K</td>
<td>5.47</td>
<td>.551 .068 (2)</td>
<td>.044 .064 .045 (2)</td>
<td>.059 .066 .054 (3)</td>
<td>.100 .098</td>
<td>.064 .083 .056 (3)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Cit-HePTh</td>
<td>citation</td>
<td>27.4K</td>
<td>704K</td>
<td>4.29</td>
<td>.562 .040 (4)</td>
<td>.031 .059 .029 (3)</td>
<td>.048 .069 .046 (3)</td>
<td>.047 .071</td>
<td>.051 .086 .044 (3)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Enron</td>
<td>communication</td>
<td>33.7K</td>
<td>362K</td>
<td>4.05</td>
<td>.615 .013 (4)</td>
<td>.010 .009 .008 (2)</td>
<td>.011 .098 .009 (4)</td>
<td>.012 .102</td>
<td>.022 .145 .012 (3)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Slashdot0902</td>
<td>social</td>
<td>82.2K</td>
<td>1.09M</td>
<td>3.94</td>
<td>.764 .048 (4)</td>
<td>.081 .052 .048 (2)</td>
<td>.085 .046 .053 (2)</td>
<td>.049 .053</td>
<td>.078 .052 .032 (4)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>DBLP</td>
<td>collaboration</td>
<td>99.3K</td>
<td>1.09M</td>
<td>3.94</td>
<td>.605 .135 (4)</td>
<td>.090 .109 .091 (2)</td>
<td>.103 .105 .099 (3)</td>
<td>.113 .096</td>
<td>.093 .102 .093 (3)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>M14B</td>
<td>electronic</td>
<td>100K</td>
<td>1.28M</td>
<td>52.5</td>
<td>1.17 .743 (3)</td>
<td>.276 .116 .267 (3)</td>
<td>.501 .152 .377 (3)</td>
<td>.270 .174</td>
<td>.059 .065 .054 (2)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Wave</td>
<td>electronic</td>
<td>156K</td>
<td>2.1M</td>
<td>22.9</td>
<td>.531 .461 (2)</td>
<td>.199 .142 .194 (3)</td>
<td>.270 .126 .211 (3)</td>
<td>.164 .120</td>
<td>.121 .079 .096 (3)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Web-Stanford</td>
<td>web</td>
<td>255K</td>
<td>3.88M</td>
<td>7.31</td>
<td>.343 .007 (2)</td>
<td>.003 .006 .003 (2)</td>
<td>.005 .007 .006 (2)</td>
<td>.007 .010</td>
<td>.010 .008 .004 (4)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Web-Google</td>
<td>web</td>
<td>856K</td>
<td>5.58M</td>
<td>6.18</td>
<td>.884 .149 (3)</td>
<td>.006 .019 .005 (3)</td>
<td>.006 .006 .005 (4)</td>
<td>.006 .009</td>
<td>.006 .010 .005 (4)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Wiki-Talk</td>
<td>communication</td>
<td>2.39M</td>
<td>9.31M</td>
<td>3.91</td>
<td>.775 .030 (4)</td>
<td>.038 .122 .038 (3)</td>
<td>.037 .040 .037 (4)</td>
<td>.039 .088</td>
<td>.036 .128 .036 (2)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>LiveJournal</td>
<td>social</td>
<td>4.00M</td>
<td>69.3M</td>
<td>5.39</td>
<td>.831 .163 (3)</td>
<td>.067 .079 .067 (2)</td>
<td>.075 .082 .071 (2)</td>
<td>.082 .079</td>
<td>.069 .074 .071 (2)</td>
<td>.000 .003 .000 (2)</td>
</tr>
<tr>
<td>Hyves</td>
<td>social</td>
<td>8.08M</td>
<td>912M</td>
<td>4.75</td>
<td>.528 .038 (3)</td>
<td>.045 .065 .035 (3)</td>
<td>.035 .055 .035 (2)</td>
<td>.034 .060</td>
<td>.042 .069 .029 (3)</td>
<td>.000 .003 .000 (2)</td>
</tr>
</tbody>
</table>

**Table:** Performance (error rate, lower is better) of different landmark selection approaches on various network datasets.
Shortest path lengths in real-world graphs can accurately be approximated using the landmark framework (error < 0.05).

A good landmark selection strategy takes into account at least:
- Selecting central nodes
- Covering different areas of the graph.

Adaptive landmark selection and GCN-processing are useful mechanisms for selecting and processing landmark candidates.

The paths can be shortened using various optimization techniques.

Finding an optimal set of landmarks will (probably) always remain a challenging task.
Bibliography


Please consult the paper for a full list of references.
Thank You!

Questions?