Landmark Based Distance Approximation Methods In Large Graphs

By

Miss Shruti Biswal
Roll No. 110CS0121
THIRD YEAR, B.TECH(COMPUTER SCIENCE & ENGG.)
DEPARTMENT OF COMPUTER SCIENCE & ENGG.
NIT ROURKELA, INDIA

UNDER THE SUPERVISION OF
Professor Cheng Hong
DEPARTMENT OF SYSTEMS ENGG. & ENGG. MANAGEMENT
THE CHINESE UNIVERSITY OF HONG KONG
HONG KONG

THESIS COMPLETED FOR THE
SUMMER INTERNSHIP PROJECT
B.TECH (COMPUTER SCIENCE & ENGG.)
NIT ROURKELA, INDIA

JULY, 2013

Professor Cheng Hong
Department of Systems Engineering
& Engineering Management
The Chinese University of Hong Kong
Shatin, Hong Kong

CERTIFICATE

This is to certify that thesis entitled **"Landmark Algorithms for Shortest Path Problem"** has been completed by *Miss Shruti Biswal, Roll No. 110CS0121, National Institute of Technology, Rourkela, India* ,during the period 6th May, 2013 - 5th July, 2013 at Department of Systems Engineering & Engineering Management,The Chinese University of Hong Kong as the requirement for the Summer Internship Project. The thesis has fulfilled the requirements of the regulations relating to the Summer Internship of the B.Tech Degree (Computer Science & Engg.) of the Institute, NIT Rourkela.

(Prof. Cheng. Hong)

Contents

Chapter 1

Introduction

1.1 Background and Motivation

The shortest path problem is a classical combinatorial optimization problem that aims to find a shortest path between two given vertices. There are many applications of this problem and many algorithms have been proposed and used in real situations. A social network can be referred to as a very large graph.It has become necessary in this recent socially-connected world, to have the shortest distance between two specific users in the network.And for this purpose, many algorithms have been proposed and devised. For a network with non-negative edge length, the Dijkstra algorithm is the most well-known algorithm which can be implemented to have the running time of O(m+ nlog n), where m and n denotes the number of edges and the number of vertices respectively.For unweighted graphs,shortest paths can be computed using Breath First Search in time O(m+n).But these being online methods are not preferred due to more time complexity as compared to other offline processes. In this thesis, the offline process to measure shortest distance between the vertices used is the landmark-embedded distance estimation method.This method has also been introduced with modifications to improve the error rate between the estimated and true distance between the vertices.

1.2 Basic Definitions

It is necessary to define and pre-determine the notations that are to be used in the forthcoming sections for implementation,application and usage purpose.

1.2.1 Notations

Let $G = (V, E)$ denote a graph having $|V| = n$ vertices and $|E| = m$ edges.For simplicity, the graphs that are considered here are undirected and unweighted graphs i.e weight of all the edges is unity. .The *distance* between two vertices s and t in a graph is the length of the shortest path between s and t and it is denoted by $d(s, t)$ where $s, t \in V$. The shortest path,denoted by $\pi_{s,t}$,is the minimum number of nodes one has to go through to reach the second node from the first one (or vice versa, undirected). So, $\pi_{s,t}$ contains the sequence of vertices that are traversed to move from s to t. So, if $\pi_{s,t}$ is a path of length d then it can be expressed as $\pi_{s,t} = \{s, v_1, v_2, .., v_{d-1}, t\}$,where $\{v_1, v_2, ..., v_{d-1}\} \subseteq V$ and$\{(s, v_1), (v_1, v_2), .., (v_{d-1}, t)\} \subseteq E$.The path length of $\pi_{s,t}$ is denoted by $|\pi_{s,t}|$. The concatenation of two paths is $\pi_{s,t}$ and $\pi_{t,u}$ is $\pi_{s,u} = \pi_{s,t} + \pi_{t,u}$.

Degree of a vertex is the number of immediate neighbors it has or the number of edges that pass through the vertex. Higher the degree of a vertex, more would be the probability of the edges to pass through that vertex.This means that most of the paths that are computed have the vertex with the maximum degree.
Landmark is a vertex which is selected from the graph $G(V, E)$ to act as an intermediate via which the distance between two vertices would be calculated.This can be explained as instead of computing a direct path between vertices s and t,where the path length is denoted as $d(s, t)$, a path via the landmark l is calculated between s and t whose distance is denoted as $d_l(s, t)$ where $s, t, l \in V$.

1.2.2 Distance Bounds

The *triangle inequality* holds true for shortest distance in a graph, since it is a metric.Given that $s, t, u \in V$,then the following inequality holds:

$$d(s,t) \leq d(s,u) + d(t,u)$$

The equality sign holds when the shortest path between s and t passes through u.

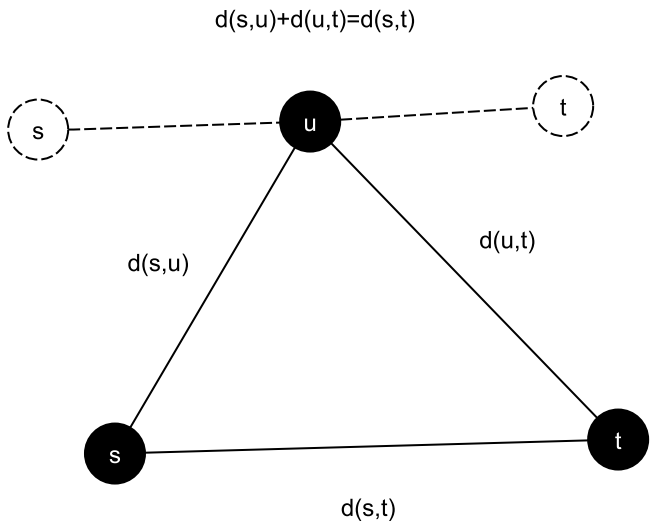

Figure 1.1: Illustrates the triangle inequality case for graph nodes

So this states that the exact distance calculated between the nodes using online methods such as Dijkstra or BFS is always greater than or equals to the approximate distance calculated between them using the landmark-based methods

Chapter 2

Algorithm Description

2.1 Introduction

In this section, we show the detail of the implementation and report the results of landmark-embedded distance estimation method. Landmarks are a few selected vertices from the graph in terms of which the shortest distance of all other vertices in the graph are measured. First, we describe the basis for selection of landmarks and its significance. And then we calculate the time taken and the approximate error obtained.Since the landmark-embedded distance estimation is an approximation method, it has a percentage of error related to it as compared to the distance that is calculated by exact methods namely Dijkstra's algorithm. Due to the less time complexity of the approximation methods these are preferred to exact methods neglecting the small relative errors.

2.2 Landmark selection

In this section, the basis of selecting landmarks is explained. Let k denote the number of landmarks we would like to choose.The landmarks can be chosen on the following basis: (i) Random vertices are selected. (ii)Highest Degree nodes are selected.

Landmarks having higher degrees have more neighbors as a result of which ,their distance to the vertices of the queries have a greater possibility of having small values and can hence generate relatively shorter distance values as compared to the landmarks selected random-

ly.The distance calculated by the landmarks selected on the basis of decreasing degrees are closer to the exact distance calculated by Dijkstra Algorithm and hence have relatively less error and hence preferred to random selection methods.

2.3 Implementation

From D = {set of k landmarks},a landmark $l \in D$ is fixed and we calculate the distance of all the vertices $v \in D$ in the graph from vertex l.This distance so calculated is denoted and stored as $d_l(v)$.
For each query (s, t) the approximate distance as measured from a landmark $l \in D$ is given as

$$d_l(s, t) = d_l(s) + d_l(t)$$

To obtain the approximate distance between the vertices in a query, distance is calculated from each of the k landmarks and the minimum is considered as the estimated distance.This is denoted as $\tilde{d}(s, t)$ as

$$\tilde{d}(s, t) = \min_{l \in D}(d_l(s, t))$$

Using Triangle Inequality, it can be shown that the approximated distance values are always greater than the exact distance.Because

$$d(s, t) < d(s, l) + d(l, t)$$

The algorithm to calculate the landmark-based approximated distance is:

Algorithm 1 :Landmark Based Estimation Method

Input: Graph $G = (V, E)$, $D = \{set\ of\ k\ landmarks\}$, $Q = \{set\ of\ queries\}$

for all $l \in D$ **do**

Do Dijkstra Algorithm to calculate distance of $v \in V$ from l and store them as $d_l(v)$ in a matrix.

end for

for all $(s, t) \in Q$ do **do**

$d_{approx}(s, t) = \min_{l \in D}(d_l(s) + d_l(t))$

end for

2.4 Improved Landmark-Based Algorithms

2.4.1 Lowest Common Ancestor Method

In this section, the performance of landmark-based method is improved by introducing a small change. Instead of using a global set of landmarks, query-specific landmarks are selected and hence distance is estimated.These query-specific landmarks are closer to the $\pi_{s,t}$ than other global landmarks.

As we elaborate, it can be seen that to calculate the approximate distance, we compute $d_l(s) + d_l(t)$, where l∈D and (s,t)∈Q .In figure 2.1, it is clear that the path from the landmark to the least-common-ancestor of s & t is added up twice and hence is subject to error.

The actual estimated distance can be expressed as

$d_{lca}(s, t) = d_l(s) + d_l(t) - 2d_l(r)$, where r= least-common-ancestor of s & t This problem arises because there is a common set of k landmarks for all the queries amongst which the minimum distance is selected.This global set of landmarks when used for computation of approximate distance may result is re-traversal of same common path twice. So in order to eliminate this error, local landmarks corresponding to each of the queries are identified using the global landmark as the origin.

For example, the shortest paths from landmark l to the query (s, t) are:

$$\pi_{l,s}(l, v_1, v_4, s) \quad and \quad \pi_{l,t}(l, v_1, v_3, v_2, t)$$

Of both, the minimum number of intermediate vertices is 3. Designating each of the intermediate vertex by a depth value, we get that : for $\pi_{l,s}$, depth 1 has v_1 and depth 2 has v_4.Likewise, for $\pi_{l,t}$, depth 1 has v_1 and so on. As we travel backwards from s and t towards l, we find that at depth 1 the first common vertex v_1 is found which is the least-common-ancestor.Pictorially,it can be represented as in Figure 2.1

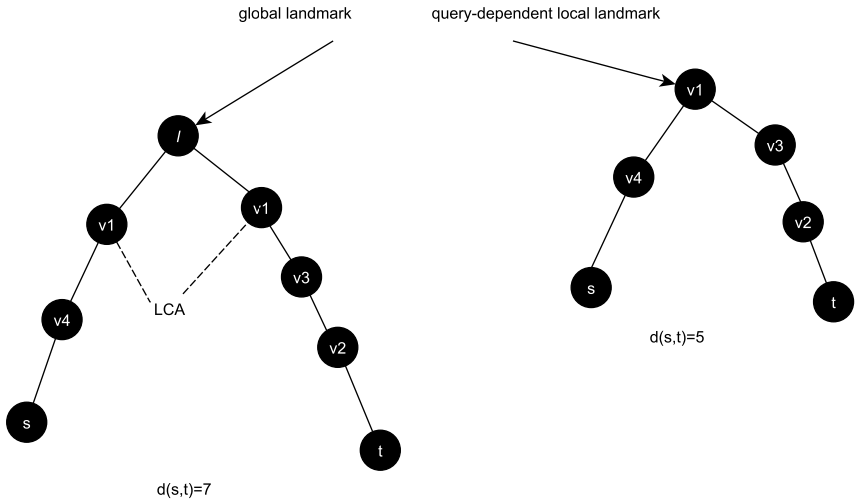

Figure 2.1: *Illustrates that the distance computed from the query dependent local landmark is shorter as computed from a global landmark.The LCA of(s,t) serves as the query dependent local landmark*

Here, we find that the if query-dependent local landmark is used for distance estimation then better results are obtained.

The algorithm to identify the LCA and calculate the improved approximate distance:

For each of the global-landmark, the shortest-path-tree to each of the vertices of a query is constructed and the least common ancestor of both the vertices is identified by traversing upwards from the vertices towards the global-landmark (origin) and checking the vertices at same depth. The first common vertex found is the least-common-ancestor (LCA) and this

Algorithm 2 :LCA Approximation Method

Input: Graph $G = (V, E)$, $D = \{set\ of\ k\ landmarks\}$, $Q = \{set\ of\ queries\}$

for all $l \in D$ **do**

Do Dijkstra Algorithm to print the paths from l to s and t where $(s, t) \in Q$ and store them as $\pi_{l,s}$ and $\pi_{l,t}$.

Add these paths to Π.

end for

for all each path in Π **do**

Find the LCA for each pair of queries and reset the paths $\pi_{l,s}$ and $\pi_{l,t}$

Compute the distance $d_l(s, t) = d_{\tilde{l}}(s) + d_{\tilde{l}}(t)$, where $d_l(s)\ and\ d_l(t)$ are the new distance of s and t to the newly found LCA,$\tilde{l}$

end for

for all $(s, t) \in Q$ **do**

$d_{approx}(s, t) = \min_{l \in D}(d_l(s) + d_l(t))$

end for

is considered as the local-landmark for that particular query. Similar procedure is carried out for all the remaining landmarks and finally the minimum distance hence obtained is the approximate distance.These local landmarks and their paths to the queries are shorter as compared to the paths traversed by the global landmarks

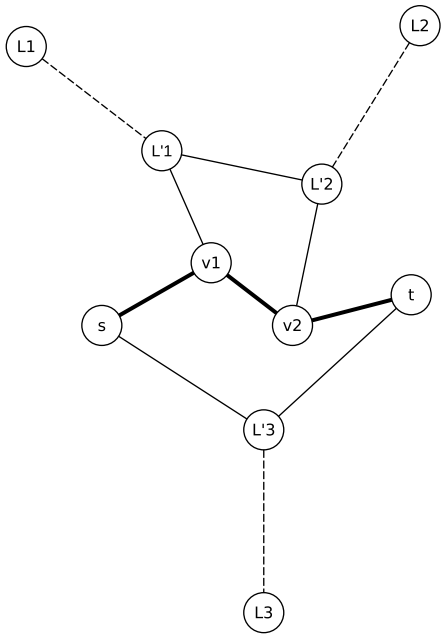

Figure 2.2: *Local landmarks have paths closer to the shortest paths as compared to global landmarks*

As in Figure 2.2, it can be clarified that $D = \{L1, L2, L3\}$ is a set of global landmarks and their paths to query (s, t) are as shown. It is also found that vertex $L'1$ is the local landmark specific to query (s, t) with respect to the global landmark $L1$.Similarly, $L'2$ and $L'3$ are local landmark to (s,t) as compared to the global landmark $L2$ and $L3$ respectively.

2.4.2 Shortcut Method

Apart from using landmark as an intermediate, there can exist a shortcut path between the two nodes of a query.This path corresponds to the vertices which are common neighbors of those nodes lying in the shortest path trees $\pi_{l,s}$ and $\pi_{l,t}$. The vertices lying on the SPT of the two nodes of the query have certain immediate neighbors. It is possible that there must be a few neighbors which are common to both the SPTs. And hence can generate a shortcut path between the two nodes and can give an alternate path between them which may have distance value less than the previous implementation methods.

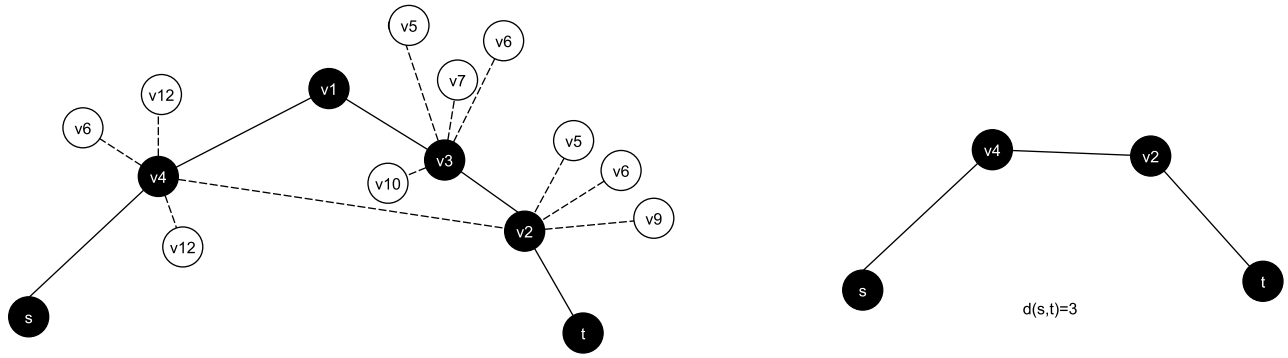

Figure 2.3: *Illustrates the neighbors of the intermediate vertices on the SPT (by dashed lines) and the shortcut path for (s,t)*

As we have seen in the example described in the previous section, the shorter paths that was found were

$$\pi_{l,s}(v_1, v_4, s) \quad and \quad \pi_{l,t}(v_1, v_3, v_2, t)$$

Here the distance between s and t is 5.
Now it is found that v_2 and v_4 share a common edge, then a shortcut path can be developed between s and t via these two nodes. Hence, the path would be shorter in length :

$$\pi_{s,t}(s, v_4, v_2, t)$$

Now the path length can be seen as 3.Since this value would be closer to the true distance this would hence generate less error and therefore, improves the LCA distance estimation method. (See Figure 2.2)
In order to locate the shortcuts, all pairs of vertices of $\pi_{l,s} \times \pi_{l,t}$ are needed to be analyzed to find the common neighbors,if any.If the pairs have any common edge, then the edge

providing the best distance estimate is considered to be the shortcut-path length which is then compared with the true distance for error calculation.

Algorithm 3 : Shortcut Paths Method

Input: Graph $G = (V, E)$, $D = \{set\ of\ k\ landmarks\}$, $Q = \{set\ of\ queries\}$

for all path in Π **do**

  for all each pair of vertices in $\pi_{l,s} \times \pi_{l,t}$ **do**

    if any common neighbor(common edge) is found **then**

      calculate distance between the two nodes of the query and store it in $d_l(s, t)$

    end if

  end for

end for

for all each query in Q **do**

  Find the best distance(shortest distance) estimated amongst all the landmarks.

end for

Chapter 3

Experimental Evaluation

3.1 Datasets

The experiment is conducted to find the shortest distance between a given pair of nodes known as query. In order to get a very large graph to carry out this experiment, a sample dataset from Facebook is obtained. Facebook is a social networking site having over billions of users. Here, the users are connected to each other as 'friends' and many are yet to be connected amongst each other.

In this dataset, the users have been assigned numbers and are the vertices,V of this very large graph. Theses vertices are connected amongst each other on the basis of the fact that they are socially connected on the website.Their connections are the edges of the graph which are denoted by E.Here, only a small section of the huge database is used as a dataset for the experiment.

With the help of proper coding it was computed that the dataset had 63731 users(vertices).So, the experiment is carried out on a graph having 63731 vertices and over a million number of edges, which is a large graph.

3.2 Experimental Setup

In order to evaluate the performance of the algorithms, first a random sample of 10000 vertex pairs are selected.Each of these pairs, (s, t) is referred to as a (query) where $u, v \in V$.

This query set is denoted by Q. The same set of queries is used for all the algorithms for performance evaluation. For each of the pair in Q, the actual distance between the nodes in the query is calculated using the Dijkstra's Algorithm. One by one, each of the approximation methods are applied to evaluate the approximate distance for each query, which is then used to calculate the approximation error and time taken to calculate these values.

Approximation error is calculated as $(d' - d)/d$ where d = exact distance calculated and d' =approximate distance calculated.

Query Time is the time taken to calculate the approximate distance between two nodes of a query. The true distance is calculated and has been represented in the following histogram to list the frequency in which distance values actually lie.

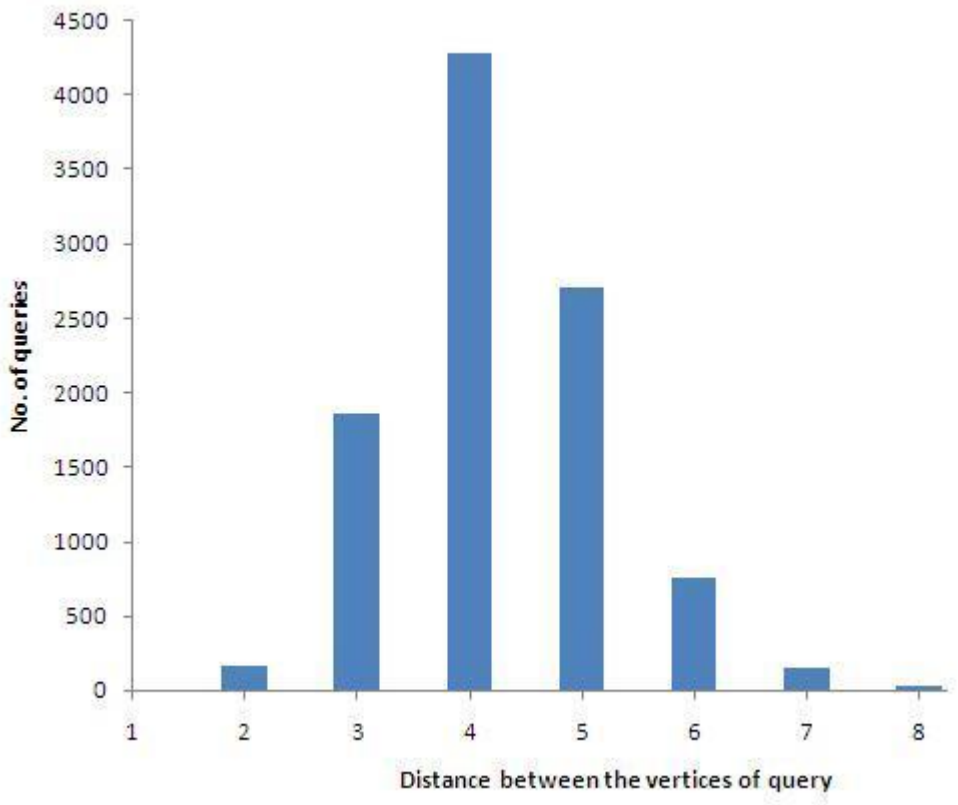

Table 3.1: Relative Approximation Error

Landmark Selection	*Methods*	k= 20	40	60	80	100
Random Landmarks	GL	0.0805	0.0602	0.0445	0.0406	0.0368
	LL	0.0127	0.0061	0.0001	0.0001	0.0001
	Shortcuts	NIL	NIL	NIL	NIL	NIL
Degree − Based Landmarks	GL	0.0097	0.0084	0.0076	0.0067	0.0059
	LL	0.0062	0.0043	0.0037	0.0031	0.0028
	Shortcuts	0.0032	0.002	0.0001	0.0001	0.0001

3.3 Results

3.3.1 Approximation Error

Implementation of the methods on the dataset reveal the difference between the true distance and the estimated distance values. Table 3.1 shows the relative error for random and degree-based selection of global landmarks and local landmarks and for the shortcut paths. For randomly selected landmarks, it is found that relative error is more in global landmarks as compared to the query dependent local landmarks. This is because local landmarks have path shorter than the global ones.But the global landmarks perform much better when the landmarks are selected on the basis of degree.

But for the local landmarks, the relative error is less in case of randomly selected landmarks because the degree based landmarks are already in their best case and hence tend to provide less improvements in the results. However,the approximation error is minimum in case of shortcut method for distance estimation. Also it is observed that the relative error decreases as the number of landmarks selected is increased i.e for k = 20 ,40, 60, 80 ,100 it is found that the relative error was decreasing for all cases.The results have been summarized and shown in Figure 1.

3.3.2 Query Time

It is observed that the query time increases as the approximation error decreases.More improvement in the method refers to more computations being carried out and hence the time taken to calculate the distance between the nodes in a query is higher.Query time for shortcut-path method is more as compared to other methods. For randomly selected landmarks the time taken by the local landmark method is about 3 times the query time in case of global landmarks for all values of k. For landmarks selected on the basis of degree, the time taken by the local landmarks is also around 3 time the global landmarks.But the shortcut method has maximum qury time as compared to other methods. This has been summarised in Table 3.2.

Table 3.2: Query Time (in ms)

Landmark Selection	*Methods*	k= 20	40	60	80	100
Random Landmarks	GL	0.021	0.038	0.06	0.076	0.1006
	LL	0.07	0.149	0.219	0.292	0.368
	Shortcuts	NIL	NIL	NIL	NIL	NIL
$Degree - Based\ Landmarks$	GL	0.017	0.037	0.057	0.073	0.093
	LL	0.06	0.119	0.182	0.247	0.31
	Shortcuts	6.493	14.696	23.559	32.746	41.249