

2009

Indexing Distances in Large Graphs and Applications in Search Tasks

Potamias, Michalis. "Indexing Distances In Large Graphs And Applications In Search Tasks (MA Thesis)", Technical Report BUCS-TR-2008-029, Computer Science Department, Boston University, December 1, 2008. [Available from: <http://hdl.handle.net/2144/1721>]

<https://hdl.handle.net/2144/1721>

"Downloaded from OpenBU. Boston University's institutional repository."



**INDEXING DISTANCES IN LARGE GRAPHS
AND APPLICATIONS IN SEARCH TASKS**

MICHALIS POTAMIAS

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Arts

**BOSTON
UNIVERSITY**

BOSTON UNIVERSITY
GRADUATE SCHOOL OF ARTS AND SCIENCES

Thesis

**INDEXING DISTANCES IN LARGE GRAPHS
AND APPLICATIONS IN SEARCH TASKS**

by

MICHALIS POTAMIAS

Diploma of Engineering, National Technical University of Athens, 2005

Submitted in partial fulfillment of the
requirements for the degree of

Master of Arts

2009

Approved by

First Reader

George Kollios, PhD
Associate Professor of Computer Science

Second Reader

Stan Sclaroff, PhD
Professor of Computer Science

Leave this world a little better than you found it.
Robert Baden-Powell

Acknowledgments

First I would like to thank my advisor, Professor George Kollios for his constant guidance and support. I would also like to thank Vassilis Athitsos and Aris Gionis who have eagerly guided me and worked with me at various times during my graduate studies. A special thank you goes to Professor Stan Sclaroff for his insightful remarks on the content of this thesis. I was also lucky to collaborate with Francesco Bonchi and Carlos Castillo during my participation in the Yahoo! Research Internship program; this work could not have been completed without their contribution.

I thank my dearest friends Giorgos, Giorgos, Niky, Panos, Vijay and Yola for putting up with me. I cannot express enough gratitude to my closest friends back home for still being around despite the distance. As always, above all, I am grateful to my family.

INDEXING DISTANCES IN LARGE GRAPHS

AND APPLICATIONS IN SEARCH TASKS

MICHALIS POTAMIAS

ABSTRACT

This thesis elaborates on the problem of preprocessing a large graph so that single-pair shortest-path queries can be answered quickly at runtime. Computing shortest paths is a well studied problem, but exact algorithms do not scale well to real-world huge graphs in applications that require very short response time.

The focus is on approximate methods for distance estimation, in particular in *landmarks-based distance indexing*. This approach involves choosing some nodes as landmarks and computing (offline), for each node in the graph its embedding, i.e., the vector of its distances from all the landmarks. At runtime, when the distance between a pair of nodes is queried, it can be quickly estimated by combining the embeddings of the two nodes. Choosing *optimal* landmarks is shown to be hard and thus heuristic solutions are employed. Given a budget of memory for the index, which translates directly into a budget of landmarks, different landmark selection strategies can yield dramatically different results in terms of accuracy. A number of simple methods that scale well to large graphs are therefore developed and experimentally compared. The simplest methods choose central nodes of the graph, while the more elaborate ones select central nodes that are also far away from one another. The efficiency of the techniques presented in this thesis is tested experimentally using five different real world graphs with millions of edges; for a given accuracy, they require as much as 250 times less space than the current approach which considers selecting landmarks at random. Finally, they are applied in two important problems arising naturally in large-scale graphs, namely social search and community detection.

Contents

1	Introduction	1
2	Related work	3
3	Algorithmic framework	7
3.1	Notation	7
3.2	Distance bounds	8
3.3	Using landmarks	8
3.4	Selecting good landmarks	10
3.5	Problem complexity and approximation algorithms	11
4	Landmarks selection heuristics	12
4.1	Basic heuristics	12
4.2	Constrained heuristics	13
4.3	Partitioning-based heuristics	13
4.4	Estimates using the lower bounds	14
5	Experimental evaluation	16
5.1	Datasets	16
5.2	Approximation quality	18
5.3	Computational efficiency	19
6	Application to Social Search	22
6.1	Problem definition	22
6.2	Evaluation method	23
6.3	Experimental results	24
7	Application to Community Detection	26

8 Conclusions and future work	28
References	29

List of Tables

- 5.1 Summary of properties of the graphs 17
- 5.2 Approximation error across datasets, using 20 (top) and 100 landmarks (bottom) . . 19
- 5.3 Indexing time. Partitioning and selecting times are expressed in wallclock seconds
for $d = 100$ landmarks in the Flickr-E dataset 20
- 6.1 Summary of precision at 5 across datasets, using 20 (top) and 100 landmarks (bottom) 24
- 7.1 Clusters obtained in DBLP. 27

List of Figures

- 3.1 Illustration of the cases for obtaining tight upper bounds (left) and tight lower bounds (right) as provided by Observations 1 and 2 8

- 5.1 Distributions of distances in all datasets. 16
- 5.2 Error of random-pair shortest paths 18

- 6.1 Precision at 5 for the social search task 25
- 6.2 Precision at 10 for the social search task 25

List of Abbreviations

APSP	All Pairs Shortest Paths
BFS	Breadth First Search
DBLP	Digital Bibliography and Library Project
Flickr-E	Flickr Explicit Graph
Flickr-I	Flickr Implicit Graph
NP	Nondeterministic Polynomial
P@k	Precision at k
SPSP	Single Pair Shortest Path
SSSP	Single Source Shortest Path
STL	Standard Template Library
VLSI	Very large Scale Integration
Y!IM	Yahoo! Instant Messenger

Chapter 1

Introduction

Understanding the mechanisms underlying complex networks' characteristics and evolution is an important task which has received interest by various disciplines including sociology, biology, and physics. With the continuously increasing availability of very large networks, the algorithmic problems that are seemingly simple become challenging tasks in practice.

One basic operation in networks is to measure how close an entity is to another, and the most intuitive distance for this is the *geodesic distance* or *shortest-path distance*. Computing shortest-path distances among nodes in a graph is an important primitive in a variety of applications including among others social behavior analysis, VLSI design in electronics, protein interaction networks in biology, and route computation in transportation. Recently, new motivating applications have arisen in the context of web search and social networks. In web search, the distance of a query's initiation point (i.e. the query *context*) to the relevant webpages could be an important aspect in the ranking of the results. In social networks, a user may be interested in finding other users, or in finding content from users that are close to her in the social graph (Singla and Richardson, 2008). This *socially sensitive* search model has been suggested as part of the social network search experience (Amer-Yahia et al., 2008).

Although computing shortest paths is a well studied problem, existing algorithms do not scale well to real-world large graphs containing millions of nodes, and tens of millions of edges, especially when the application requires an answer time in the order of milliseconds. For example, Breadth-First Search (BFS) traversal of a graph of 5M nodes and 50M edges takes roughly a minute in a standard modern desktop computer. Precomputing all the shortest paths and storing them explicitly is infeasible: one would need to store a matrix of 12.5 trillion elements.

The methods described in this thesis use precomputed information to provide with fast estimates or bounds of the actual distance in very short time (i.e. milliseconds). The offline step

consists of choosing some nodes as *landmarks* (also referred to as *reference objects*) and computing distances from every node to them. This kind of precomputed information is often referred to as an *embedding*.

Contribution. In this thesis an extensive analysis of various heuristics for selecting landmarks is presented. More than 30 simple heuristics that scale well to very large networks were devised and experimentally compared. For presentation sake, only the most significant ones are reported. The experimentation shows that for a given target accuracy, these techniques require far less space than random landmark selection, allowing an efficient approximate computation of shortest path distances among nodes in very large graphs. To our knowledge, this is the first systematic analysis of landmarks selection strategies for shortest path computation in large graphs until now. The main contributions can be summarized as follows:

- The problem of optimal landmark selection in a graph is defined and proven to be **NP**-hard (Chapter 3).
- Simple and intuitive heuristics to choose landmarks that scale well to huge graphs are proposed (Chapter 4).
- The effectiveness and robustness of these techniques are tested experimentally using five large different real-world datasets (Chapter 5).
- Application to search in four social networks and a webgraph, shows high precision and low error in the problem of finding the closest nodes in the graph that match a given query (Chapter 6).
- The precomputed embeddings can be used for fast and meaningful graph partitioning (Chapter 7).

Experimental evaluation is conducted using real world networks: social graphs with explicit or implicit links from Flickr (a popular photo sharing portal), a graph based on the communication network of the Yahoo! Instant Messenger service, and the coauthors graph from the DBLP records. A web-graph defined by the Wikipedia pages and their hyperlinks is also used.

Chapter 2

Related work

This section outlines the related work in exact and approximate computation of shortest-path distances on a graph with n vertices and m edges.

Exact shortest-path distances. Dijkstra described the algorithm to compute single source shortest paths (SSSP) in weighted graphs from a node to all others (Dijkstra, 1959). The cost is $O(n^2)$ in general and can be reduced to $O(m + n \log n)$ for sparse graphs. For unweighted graphs shortest paths can be computed using Breadth First Search (BFS) in time $O(m + n)$. The Floyd-Warshall algorithm employs dynamic programming to solve the all-pairs shortest paths (APSP) problem in an elegant and intuitive way (Floyd, 1962) in time $O(n^3)$. Still the complexity of computing APSP by invoking n Dijkstra/BFS computations is asymptotically faster, since it costs $O(nm + n^2 \log n)$ and $O(nm)$ respectively.

Goldberg et al. (Goldberg, 2007) address the problem of single-pair shortest-path (SPSP) and employ landmarks in order to prune the search space of the shortest path computation. Their landmarks are similar to the ones used for the experiments of this thesis for the lower-bound estimation (Section 4.4); instead in this thesis heuristics for selecting landmarks that work well with upper-bound estimates are presented. This thesis addresses a different problem since it elaborates not on the shortest path itself, but only on the length of the shortest path. The proposed algorithms work only on the precomputed node-to-landmark-distances and do not perform any Dijkstra-type computation at query-time.

Indexing for approximate shortest-paths. The main interest is on preprocessing a graph so that SPSP queries can be answered approximately and quickly at runtime. Thorup and Zwick (Thorup and Zwick, 2001) observe that this problem is probably the most natural formulation of the APSP. In their paper they obtain the result that for any integer $k \geq 1$ a graph can be preprocessed in

$O(kmn^{\frac{1}{k}})$ expected time, constructing a data structure of size $O(kn^{1+\frac{1}{k}})$, and a SPSP query can be processed in time $O(k)$. The quotient of the division of the estimated distance and the exact is guaranteed to lie within $[1, 2k - 1]$. For $k = 1$ we get the straightforward exact solution: compute all shortest paths and store them. The latter costs $O(nm)$ time and $O(n^2)$ space. For large sparse graphs such as the ones considered in this thesis, even if one could afford the computational cost of APSP, the space cost would be quadratic to the number of nodes, which is always much larger than the one needed to store the graph itself.

For $k = 2$ the estimate may be three times more than the actual distance. In large real-world graphs this bound is already not informative due to the *small-world phenomenon*; in a scale-free network such as the explicit Flickr-contacts graph described in Section 5, for an estimated distance of 6, the exact distance is only guaranteed to lie within the interval $[2, 6]$, along with almost every pairwise distance. A survey on exact and approximate distances in graphs can be found in (Zwick, 2001).

Embedding methods. The work reported in this thesis is also strongly related to general embedding methods. In domains with a computationally expensive distance function, significant speedups can be obtained by embedding objects into another space and employing a more efficient distance measure. Several methods have been proposed to embed a space into a Euclidean space (Bourgain, 1985; Hjalton and Samet, 2003). There have been attempts to optimize the selection of reference objects for such a setting (Athitsos et al., 2008; Venkateswaran et al., 2006).

Landmarks have already been used for internet measurements (Dabek et al., 2004; Ng and Zhang, 2002; Tang and Crovella, 2003). Rattigan et al. combine their *zone* approach with the landmark technique in order to measure shortest paths in general graphs and finally to estimate centrality measures. The work presented in this thesis is complementary to these techniques. Kleinberg et al. discuss the problem of approximating network distances in real networks (not necessarily shortest paths) via embeddings using a small set of *beacons* (i.e., landmarks) (Kleinberg et al., 2004). Of most interest is the fact that they introduce in their analysis the notion of *slack*, as a quantity of pairs in the network for which the algorithm provides no guarantees. Their analysis considers choosing beacons randomly. In this thesis we show that in practice, simple intuitive heuristics work much

better than the random. Abraham et al. generalize the metric embedding with slack (Abraham et al., 2005).

Applications. Several measures have been introduced to measure the *centrality* of a vertex. This work is tightly connected to these notions. Betweenness centrality measures the amount of shortest paths passing from a vertex while closeness centrality measures the average distance of a vertex to any other vertex in the network (Freeman, 1977). Brandes gave the best known algorithm to compute the exact *betweenness centrality* of all vertices by adapting the APSP Dijkstra algorithm (Brandes, 2001). The algorithm runs in $O(nm + n^2 \log n)$ time, which is prohibitive for large graphs. Bader et al. gave a sampling-based approximation algorithm (Bader et al., 2007) and showed that centrality is easier to approximate for central nodes.

Closeness centrality is used in this thesis as a heuristic for choosing central points as landmarks in the graph.

Fast SPSP computation is becoming very relevant to Information Retrieval. Socially sensitive search in social networks and initiating-location-sensitive-search are attracting remarkable interest in the information retrieval community (Baeza and Ribeiro, 1999). It has been reported that people who chat with each other are more likely to share various interests (Singla and Richardson, 2008). An experiment discussed thoroughly in Section 5 considers ranking search results in social networks based on shortest path distances. This problem has also been studied recently by Vieira et al. (Vieira et al., 2007). Their work is also based on landmarks, but their landmarks are chosen randomly. Recently Amer-Yahia et al. consider network-aware search in collaborative tagging sites (Amer-Yahia et al., 2008). Even though their technique does not use shortest-path distance, it does take into account the neighborhood of a node. In the web-search filed, Kraft et al. suggest that a query in web-search can be enriched with context from its source, i.e., the page the user was browsing at the time of the query initiation (Kraft et al., 2006). Recently Ukkonen et al. used this idea as a component of the ranking function in searching within Wikipedia (Ukkonen et al., 2008). The growing interest in involving context and/or social connections in searching tasks suggests that distance computation will soon be a primitive of ranking functions.

One of the most studied problems in social networks is the identification of “communities”, a concept referring to the fact that nodes in many real networks appear to group in subgraphs in

which the density of internal connections is larger than the connections with the rest of nodes in the network. Community detection has received interest in physics and sociology (Freeman, 1993; Girvan and Newman, 2002; Newman and Girvan, 2004; Wasserman and Faust, 1994). More recently, with the increasing availability of very large graphs, computer scientists have started studying community detection in the web (Gibson et al., 1998; Kumar et al., 1999), and in social networks (Backstrom et al., 2006; Hopcroft et al., 2003; Kumar et al., 2006). In Section 7 it is suggested that, when landmarks are chosen appropriately, node-embeddings may contain rich information about the nodes' position within the graph. By using the embeddings as the objects features, and adopting a standard Euclidean distance between embeddings, we can reduce the community detection task to a standard clustering on relational data.

Chapter 3

Algorithmic framework

In this section the notation that is used in the rest of the thesis is introduced. A description of how to index distances very efficiently using landmarks is also given. The landmark-selection problem, considered in this thesis, is formally defined and proven **NP**-hard .

3.1 Notation

Consider a graph $G(V, E)$ with n vertices and m edges. Given two vertices $s, t \in V$, define $\pi_{s,t} = \langle s, u_1, u_2, \dots, u_{l-1}, t \rangle$ to be a path of distance $|\pi_{s,t}| = \ell$ between s and t , if $\{u_1, \dots, u_l\} \subseteq V$ and $\{(s, u_1), (u_1, u_2), \dots, (u_{l-1}, t)\} \subseteq E$, and let $\Pi_{s,t}$ be the set of all paths from s to t . Accordingly, let $d_G(s, t)$ be the distance corresponding to the shortest path between any two vertices $s, t \in V$. In other words, $d_G(s, t) = |\pi_{s,t}^*| \leq |\pi_{s,t}|$ for all paths $\pi_{s,t} \in \Pi_{s,t}$. Let $\text{SP}_{s,t}$ be the set of paths whose length is equal to $d_G(s, t)$.

For simplicity consider unweighted, undirected graphs, but all the ideas in this thesis can be easily applied to weighted and/or directed graphs.

Consider an ordered set of d vertices $D = \langle u_1, u_2, \dots, u_d \rangle$ of the graph G , which we call *landmarks*. The main idea is to represent each other vertex in the graph as a vector of shortest path distances to the set of landmarks. This is also called an *embedding* of the graph. In particular, each vertex $v \in V$ is represented as d -dimensional vector $\phi(v)$:

$$\phi(v) = \langle d_G(v, u_1), d_G(v, u_2), \dots, d_G(v, u_d) \rangle \quad (3.1)$$

For ease of presentation, from now on the i -th coordinate of $\phi(v)$ is denoted by v_i , i.e., $v_i = d_G(v, u_i)$.

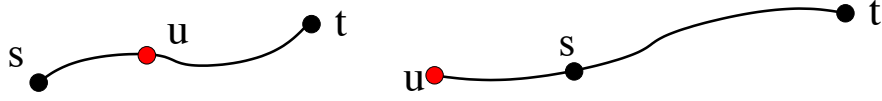


Figure 3-1: Illustration of the cases for obtaining tight upper bounds (left) and tight lower bounds (right) as provided by Observations 1 and 2

3.2 Distance bounds

The shortest-path distance in graphs is a metric, and therefore it satisfies the triangle inequality. That is, given any three nodes s , u , and t , the following inequalities hold.

$$d_G(s, t) \leq d_G(s, u) + d_G(u, t), \quad (3.2)$$

$$d_G(s, t) \geq |d_G(s, u) - d_G(u, t)| \quad (3.3)$$

An important observation that will be used to formulate the landmark-selection problem is that if u belongs to one of the shortest paths from s to t , then the inequality (3.2) holds with equality.

Observation 1. *Let s, t, u be vertices of G . If there exists a path $\pi_{s,t} \in \text{SP}_{s,t}$ so that $u \in \pi_{s,t}$ then $d_G(s, t) = d_G(s, u) + d_G(u, t)$.*

A similar condition exists for the inequality (3.3) to be tight, but in this case, it is required that either s or t are the “middle” nodes.

Observation 2. *Let s, t, u be vertices of G . If there exists a path $\pi_{s,u} \in \text{SP}_{s,u}$ so that $t \in \pi_{s,u}$, or there exists a path $\pi_{t,u} \in \text{SP}_{t,u}$ so that $s \in \pi_{t,u}$, then $d_G(s, t) = |d_G(s, u) - d_G(u, t)|$.*

The situation described in Observations 1 and 2 is shown in Figure 3-1.

3.3 Using landmarks

Given a graph G with n vertices and m edges, and a set of d landmarks D , we precompute the distances between each vertex in G and each landmark. The cost of this offline computation is d BFS traversals of the graph: $O(md)$.

Recall that the task considered for this thesis is to compute an estimate of $d_G(s, t)$ for any two vertices $s, t \in V$. Due to Inequalities (3.2) and (3.3), we have

$$\max_i |s_i - t_i| \leq d_G(s, t) \leq \min_j \{s_j + t_j\}.$$

In other words, the true distance $d_G(s, t)$ lies in the range $[L, U]$, where $L = \max_i |t_i - s_i|$ and $U = \min_j \{s_j + t_j\}$. Notice that one landmark may provide the best lower bound and another the best upper bound. Any value in the range $[L, U]$ can be used as an estimate $\tilde{d}(s, t)$ for the real value of $d_G(s, t)$. Some choices include using the upper bound

$$\tilde{d}_u(s, t) = U,$$

using the lower bound

$$\tilde{d}_l(s, t) = L,$$

or the middle point

$$\tilde{d}_m(s, t) = \frac{L + U}{2}.$$

Notice that in all cases the estimation is very fast, as only $O(d)$ operations need to be performed, and d can be thought of as being a constant, or a logarithmic function of the size of the graph.

We found out experimentally, that the “upper bound” estimates $\tilde{d}_u(s, t) = U$ work much better than the other two types of estimates, so for the rest of this thesis the focus is on the upper-bound estimates. There is a brief comment about lower-bound estimates later in Section 4.4.

As follows by Observation 1, the approximation $\tilde{d}_u(s, t)$ is exact if there exists a landmark in D that belongs to a shortest path between s and t . This motivates the definition of *coverage*:

Definition 1. *We say that a set of landmarks D covers a specific pair of vertices (s, t) if there exists at least one landmark in D that lies in one shortest path from s to t .*

Our landmark-selection problem is formulated as follows.

Problem 1 (LANDMARKS $_d$). *Given a graph $G = (V, E)$ select a set of d landmarks $D \subseteq V$ so that the number of pairs of vertices $(s, t) \in V \times V$ covered by D is maximized.*

A related problem is the following

Problem 2 (LANDMARKS-COVER). *Given a graph $G = (V, E)$ select the minimum number of landmarks $D \subseteq V$ so that all pairs of vertices $(s, t) \in V \times V$ are covered.*

3.4 Selecting good landmarks

To obtain some intuition about landmark selection, consider the LANDMARKS_d problem for $d = 1$. The best landmark to select is a vertex that it is very *central* in the graph, and many shortest paths pass through it. In fact, selecting the best landmark is related to finding the vertex with the highest *betweenness centrality* (Freeman, 1977).

Recall the definition of Betweenness centrality. Given two vertices s and t , let σ_{st} denote the number of shortest paths from s to t . Also let $\sigma_{st}(u)$ denote the number of shortest paths from s to t that some $u \in V$ lies on. The betweenness centrality of the vertex u is then defined as follows.

$$C_B(u) = \sum_{s \neq u \neq t \in V} \frac{\sigma_{st}(u)}{\sigma_{st}} \quad (3.4)$$

The fastest known algorithms to compute betweenness centrality exactly are described by Brandes (Brandes, 2001). They extend well-known all-pairs-shortest-paths algorithms (Cormen et al., 2001). The time cost is $O(nm)$ for unweighted graphs and $O(nm + n^2 \log n)$ for weighted graphs. Additionally, Bader et al. (Bader et al., 2007) discuss how to approximate betweenness centrality by random sampling.

For our problem, we consider a modified definition of betweenness centrality according to which we define $I_{st}(u)$ to be 1 if u lies on at least one shortest path from s to t , and 0 otherwise. We then define

$$C(u) = \sum_{s \neq u \neq t \in V} I_{st}(u). \quad (3.5)$$

It follows immediately that the optimal landmark for the LANDMARKS_d problem with $d = 1$ is the vertex that maximizes $C(u)$. We note that our modified version $C(u)$ can be computed as efficiently as $C_B(u)$ with a straightforward modification of Brandes' algorithm (Brandes, 2001).

3.5 Problem complexity and approximation algorithms

Both of the problems LANDMARKS_d and LANDMARKS-COVER are **NP**-hard. An easy reduction for the LANDMARKS-COVER problem can be obtained from the VERTEX-COVER problem.

Theorem 1. LANDMARKS-COVER is **NP**-hard.

Proof. We consider the decision version of the problems LANDMARKS-COVER and VERTEX-COVER . The latter problem is defined as follows: given a graph G , and an integer k , decide if there is a subset of vertices $V' \subseteq V$ of size at most k so that for all edges $(u, v) \in E$ it is either $u \in V'$ or $v \in V'$. Transform an instance of VERTEX-COVER to an instance of LANDMARKS-COVER . Consider a solution D for LANDMARKS-COVER . Consider now the set of all 1-hop neighbors and observe that each pair is connected by a unique shortest path of length 1 (i.e. an edge). Since all pairs of vertices are covered, so are 1-hop neighbors, therefore the edges of E are also covered by D , therefore, D is a solution to VERTEX-COVER . Conversely, consider a solution V' for VERTEX-COVER . Consider a pair of vertices $(s, t) \in V \times V$, and any shortest path $\pi_{s,t}$ between them. Some vertices of V' should be on the edges of the path $\pi_{s,t}$, and therefore V' is also a solution to LANDMARKS-COVER . \square

As a consequence, LANDMARKS_d is also **NP**-hard.

Next we describe a polynomial-time approximation solution to the landmark-selection problem. The main idea is to map the problem to SET-COVER problem. Given the graph $G = (V, E)$, consider a set of elements $U = V \times V$ and a collection of sets \mathcal{S} , so that each set $S_v \in \mathcal{S}$ corresponds to a vertex $v \in V$. A set S_v contains an element $(s, t) \in U$ if v lies on a shortest path from s to t . Then by solving the SET-COVER problem on (U, \mathcal{S}) by the greedy algorithm (Chvatal, 1979) we obtain a $O(\log n)$ -approximation to LANDMARKS-COVER problem and a $(1 - 1/e)$ -approximation to LANDMARKS_d problem. However, the running time of the above approximation algorithm is $O(n^3)$, which is unacceptable for the size of graphs that we consider in this thesis.

In the next chapter heuristic algorithms for the landmark-selection problem are presented. These algorithms are very efficient and scale well for extremely large graphs. Our heuristics are motivated by the observations we made in this section regarding properties of good landmarks.

Chapter 4

Landmarks selection heuristics

This section describes the proposed landmark-selection strategies and Section 5 evaluates them experimentally.

The baseline heuristic, which has been used by many researchers in the literature, is to select landmarks at random (Vieira et al., 2007; ?; ?). The heuristics proposed in this chapter are motivated by the discussion in the previous section. On a high level, the idea is to select as landmarks “central” nodes of the graph, so that many shortest paths are traversing those landmarks. Two proxies are used for selecting central nodes: (i) high-degree nodes and (ii) nodes with high *closeness centrality*, where the closeness centrality of a node u is defined as the average distance $\frac{1}{n} \sum_v d_G(u, v)$ of u to other nodes in the graph.

Intuitively, for selecting a set of landmarks that cover many different pairs of nodes, we seek to *spread* the landmarks at different positions of the graph. To capture this intuition two modifications of the previous heuristics are proposed: (i) a *constrained* variant, landmarks that are nearby are not selected, and (ii) a *partitioning* variant, where we first partition the graph and then select landmarks from different partitions.

4.1 Basic heuristics

RANDOM: The baseline landmark-selection heuristic consists of sampling a set of d nodes uniformly at random from the graph.

DEGREE: We sort the nodes of the graph by decreasing degree and we choose the top d nodes. Intuitively, the more connected a node is, the higher the chance that it participates in many shortest paths.

CENTRALITY: We select as landmarks the d nodes with the highest closeness centrality. The intuition is that the closer a node appears to the rest of the nodes the bigger the chance that it is part of many shortest paths.

Computing the closeness centrality for all nodes in a graph is an expensive task, so in order to make this heuristic scalable to very large graphs, we resort to computing centralities approximately. The approximation works by selecting a sample of random seed nodes, performing a BFS computation from each of those seed nodes, and recording the distance of each node to the seed nodes. Since the seeds are selected uniformly at random and assuming that graph distances are bounded by a small number (which is true since real graphs typically have small diameter), the Hoeffding inequality (Hoeffding, 1963) can be employed to show that a good approximation to centrality may be obtained by sampling a constant number of seeds.

4.2 Constrained heuristics

The goal is to cover as many pairs as possible. Using a basic heuristic as the ones described above, it may happen that the second landmark we choose covers a set of pairs that is similar to the one covered by the first, and thus its contribution to the cover is small.

The constrained variants of the heuristics depend on a depth parameter h . First the nodes are ranked according to some heuristic (e.g., highest degree or closeness centrality). Then landmarks are selected iteratively according to their rank. For each landmark l selected, we discard from consideration all nodes that are at distance h or less from l . The process is repeated until d landmarks have been selected.

The modified heuristics are denoted by DEGREE/h and $\text{CENTRALITY}/h$.

For all the experiments of the next chapter $h = 1, 2, 3$ has been used and the best results have been obtained for $h = 1$; for the rest of this thesis only this latter case is considered.

4.3 Partitioning-based heuristics

To spread the landmarks across different positions of the graph, *partitioning* is employed using a graph-partitioning algorithm (such as MeTis (Karypis and Kumar, 1995)) and then landmarks are selected from the different partitions. The following partitioning-based heuristics are studied.

DEGREE/P: Pick the node with the highest degree in each partition.

CENTRALITY/P: Pick the node with the highest centrality in each partition.

BORDER/P: Pick nodes close to the *border* of each partition. We do so by picking the node u with the largest $b(u)$ in each partition, according to the following formula:

$$b(u) = \sum_{j \in C, u \in C(i), i \neq j} d_j(u) \cdot d_i(u), \quad (4.1)$$

where $C(i)$ is partition i and $d_i(u)$ is the number of neighbors of u that lie in partition i . The intuition of the above formula is that if a term $d_j(u) \cdot d_i(u)$ is large, then node u lies potentially among many paths from nodes s in partition i to nodes t in partition j : such (s, t) pairs of nodes have distance at most 2, and since they belong to different partitions, it is likely that there are no direct edges for most of them.

Extensive experimentation was conducted with partitioning heuristics that use the constrained process of the previous section (partition, select a node, remove it from the graph, then pick the next node), but they did not show improvement over simpler heuristics and thus are omitted.

4.4 Estimates using the lower bounds

As mentioned in Section 3.3, one can also use the values $\tilde{d}_l(s, t)$ and $\tilde{d}_m(s, t)$ for obtaining estimates for the shortest path length $d_G(s, t)$.

Following Observation 2, landmarks that give good lower-bound estimates $\tilde{d}_l(s, t)$ are nodes on the “periphery” of the graph, so that many graph nodes are on a shortest path between those landmarks and other nodes. Most of the heuristics discussed above are optimized to give good upper-bound landmarks, by selecting central nodes in the graph, and they perform very poorly for lower-bound landmarks.

In fact, random landmarks perform better than any of the above methods with respect to lower bounds. With the intuition to select landmarks on the periphery of the graph, the following algorithm was applied: (i) select the first landmark at random (ii) iteratively perform a BFS from the last selected landmark and select the next landmark that is the farthest away from all selected landmarks so far (e.g., maximizing the minimum distance to a selected landmark). This algorithm yields better

lower bound performance than selecting landmarks at random, but overall the performance is still much worse than any of the methods for upper-bound landmarks.

Chapter 5

Experimental evaluation

5.1 Datasets

The accuracy and efficiency of the proposed methods is demonstrated using five real-world datasets. The first four are anonymized datasets obtained from various sources, namely Flickr, Yahoo! Instant Messenger (Y!IM), and DBLP. Experiments are also conducted using a document graph from the Wikipedia (nodes are articles, edges are hyperlinks among them). The distance distributions of the datasets are presented in Figure 5-1. Next we provide more details and characteristics of these datasets.

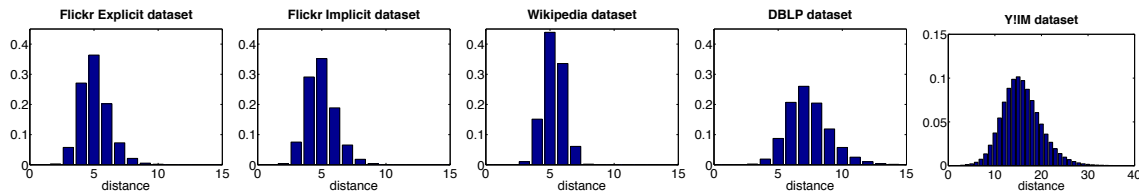


Figure 5-1: Distributions of distances in all datasets.

Flickr-E: Explicit contacts in Flickr. Flickr is a popular online-community for sharing photos, with millions of users. The first graph used in this chapter is representative of its social network, in which the node set V represents users, and the edge set E is such that $(u, v) \in E$ if and only if a user u has added user v as his/her contact.

The sample of Flickr used has 25M users and 71M relationships. In order to create a sub-graph suitable for our experimentation we perform the following steps. First, a graph is created from Flickr by taking all the contact relationships that are reciprocal. Then all users are discarded except for the ones from the US, UK, and Canada. Finally, only the largest connected component of this graph is kept.

Table 5.1: Summary of properties of the graphs

Dataset	$ V $	$ E $	$\ell_{0.9}$	$\ell_{0.5}$	c	θ	t_{BFS}
Flickr-E	588K	11M	7	6	0.15	2.0	14s
Flickr-I	800K	18M	7	6	0.11	1.9	23s
Wikipedia	4M	49M	7	6	0.10	2.9	71s
DBLP	226K	1.4M	10	8	0.47	2.7	1.8s
Y!IM	94K	265K	22	16	0.12	3.2	<1s

$\ell_{0.9}$: effective diameter, $\ell_{0.5}$: median diameter, θ : power-law coefficient, c : clustering coefficient, t_{BFS} cpu-time in seconds of a breadth-first search.

Flickr-I: Implicit contacts in Flickr. This graph models more active user behavior and user relationships. In particular, implicit relationships are inferred between Flickr users using *reciprocal comments* as a proxy for shared interest. In this graph an edge $(u, v) \in E$ exists if and only if a user u has commented on a photo of v , and v has commented on a photo by u . Again, only the largest connected component of the resulting graph is considered.

DBLP coauthor graph. The DBLP coauthors are extracted from a graph of a recent snapshot of the DBLP database that considers only the journal publications. There is an undirected edge between two authors if they have coauthored a journal paper. Only the largest connected component of this graph is considered.

Yahoo! IM graph. This is a subgraph of the Yahoo! Instant Messenger contact graph, containing users who are active also in Yahoo! Movies. Goyal et al. describe this dataset in more detail (Goyal et al., 2008). The largest connected component of the graph is kept.

Wikipedia hyperlinks. In contrast with the previous four datasets, which are social graphs, this graph is an example of a web-graph, the Wikipedia link graph. This graph represents Wikipedia pages that link to one another. All hyperlinks are undirected edges. Pages having more than 500 hyperlinks are removed, since they are mostly lists.

Summary statistics about these datasets are presented in Table 5.1. The statistics include the effective diameter $\ell_{0.9}$ and the median diameter $\ell_{0.5}$, which are the minimum shortest-path distances at which 90% and 50% of the nodes are found respectively, and the clustering coefficient c . In these graphs the degree follows a Zipfian distribution in which the probability of having degree x is proportional to $x^{-\theta}$; the parameter θ fitted using Hill’s estimator (Hill, 1975) is shown in the table.

5.2 Approximation quality

We measure the accuracy of our methods in calculating shortest paths between pairs of nodes. For this experiment we randomly choose 500 random pairs of nodes. In the case of the RANDOM selection heuristic we average the results over 10 runs for each landmark set size. For each method and dataset the average of the approximation error is reported: $|\hat{\ell} - \ell|/\ell$ where ℓ is the actual distance and $\hat{\ell}$ the approximation.

Figure 5-2 shows representative results for two datasets. Observe that using two landmarks chosen with the CENTRALITY heuristic in the Flickr-E dataset yields an approximation equal to the one provided by using 500 landmarks selected by RANDOM. In terms of space and query-time this results to savings of a factor of 250. For the DBLP dataset the respective savings are of a factor greater than 25.

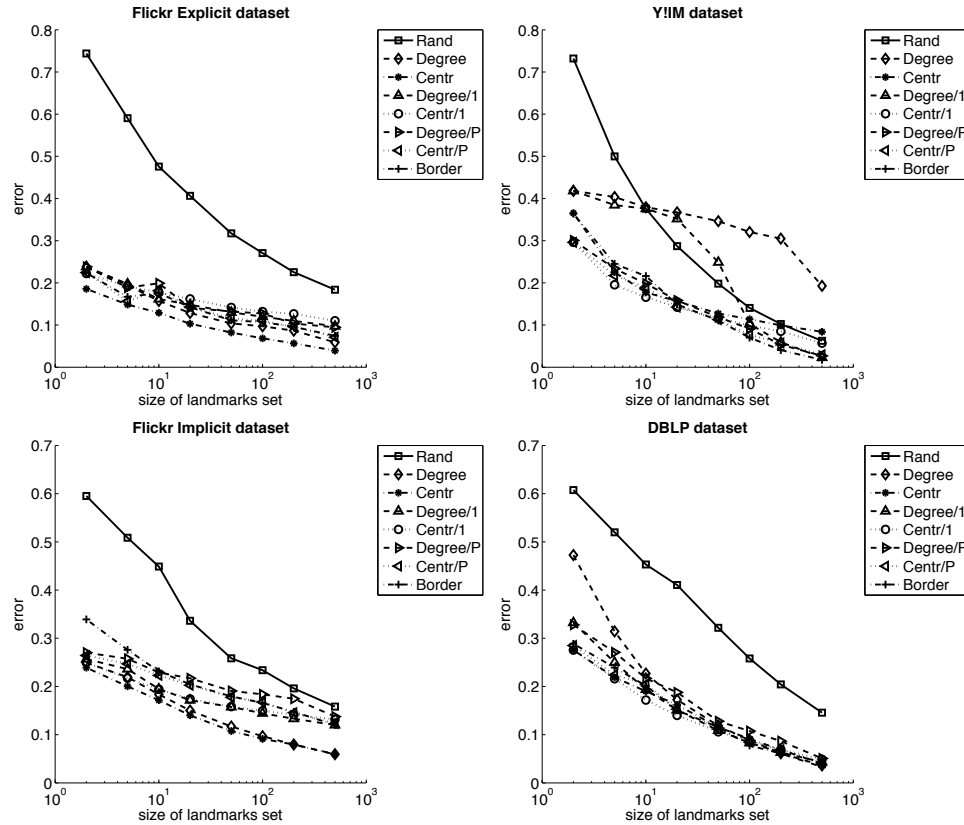


Figure 5-2: Error of random-pair shortest paths

Table 5.2 summarizes the approximation error of the heuristics across all 5 datasets studied here. We are using two landmark sizes: 20 and 100 landmarks; with 100 landmarks it is possible to observe error rates of 10% or less across most datasets.

Table 5.2: Approximation error across datasets, using 20 (top) and 100 landmarks (bottom)

20 landmarks	Fl.-E	Fl.-I	Wiki	DBLP	Y!IM
RANDOM	0.41	0.34	0.69	0.41	0.29
DEGREE	0.13	0.15	0.35	0.17	0.37
CENTRALITY	0.10	0.14	0.21	0.16	0.16
DEGREE/1	0.15	0.17	0.35	0.15	0.35
CENTRALITY/1	0.16	0.17	0.21	0.14	0.14
DEGREE/P	0.14	0.22	0.40	0.13	0.16
CENTRALITY/P	0.14	0.20	0.22	0.15	0.14
BORDER/P	0.14	0.20	0.29	0.15	0.15
100 landmarks	Fl.-E	Fl.-I	Wiki	DBLP	Y!IM
RANDOM	0.27	0.23	0.61	0.26	0.14
DEGREE	0.10	0.10	0.22	0.09	0.32
CENTRALITY	0.07	0.09	0.16	0.09	0.11
DEGREE/1	0.12	0.14	0.23	0.08	0.10
CENTRALITY/1	0.13	0.15	0.16	0.09	0.10
DEGREE/P	0.13	0.18	0.26	0.11	0.09
CENTRALITY/P	0.11	0.17	0.16	0.08	0.08
BORDER/P	0.11	0.17	0.20	0.08	0.07

By examining Table 5.2 one can conclude that even simple strategies are much better than random landmark selection. Selecting landmarks by DEGREE is a good strategy, but did not perform well in Y!IM graph. The strategies based on CENTRALITY and BORDER yield good results across all datasets.

5.3 Computational efficiency

The methods were implemented in C++ using Boost and STL Libraries. All the experiment are run on a Linux server with 8 1.86GHz Intel Xeon processors and 16GB of memory.

The method is extremely efficient at query-time. The online step is constant, $O(d)$ per pair where d is the number of landmarks, a small constant. This translates to a few milliseconds per query. The reader may compare that to the BFS time in Table 5.1 which is prohibitive for online applications. These facts support the initial claim that there are remarkable trade-offs of efficiency vs. accuracy using heuristically selected landmarks.

The running times for the offline computation are shown on Table 5.3, in some cases they depend on the time it takes to perform a BFS in each dataset, shown in Table 5.1. The offline computation includes four phases:

1. A centrality computation is required for the methods based on CENTRALITY, and it takes S BFSs in which S is the sample size of the initial seed nodes.
2. A partition of the graph is required for the methods based on partitioning */P, and in the case of the Flickr-E dataset (588K nodes, 11M edges) it takes around 30 seconds using the standard clustering method of the Metis-4.0 package.¹
3. The selection of the landmarks depends on the heuristic, but it takes between 1 and 4 seconds in the Flickr-E dataset.
4. The embedding implies labelling each node in the graph with its distance to the landmarks.

Table 5.3: Indexing time. Partitioning and selecting times are expressed in wall-clock seconds for $d = 100$ landmarks in the Flickr-E dataset

Method	Centrality [t_{BFS}]	Partition [sec]	Select [sec]	Embed [t_{BFS}]
RANDOM	-	-	<1	d
DEGREE	-	-	<1	d
CENTRALITY	S	-	<1	d
DEGREE/1	-	-	<1	d
CENTRALITY/1	S	-	<1	d
DEGREE/P	-	<30	4	d
CENTRALITY/P	S	<30	4	d
BORDER/P	-	<30	4	d

The computational time during the indexing is dominated by the BFS traversals of the graph. S such traversals are necessary for performing centrality estimations in the algorithms, basically by picking S seed nodes uniformly at random and then doing a BFS from those nodes; the centrality of a node is then estimated as its average distance to the S seeds. The embedding always takes d BFS's to be computed. These traversals can be sped-up by doing the traversals in parallel in several machines. Observe that an exact-online solution which requires a BFS/Dijkstra traversal of the graph, cannot be straightforwardly parallelized.

¹<http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>

With respect to the memory requirements to store the index, in all of the datasets more than 99% of the pairs are at a distance of less than 63, meaning that one can safely use six bits per landmark and node to store the embeddings. With 20 landmarks in a large graph of 100 M nodes, one would use 120 bits (15 bytes) per node. Thus, around 1.5 GB of memory would be required to store all the embeddings, which is a very small memory footprint for this application.

Chapter 6

Application to Social Search

In this section an application of the suggested method to network-aware search is studied. In network-aware search, the results to a search are nodes in a graph, and the originating query is represented by a *context node* in the graph. The context node may represent the user issuing the query, or a document the user was browsing when she issued the query.

Nodes that match the query are ranked using a ranking function that considers, among other factors, their relationship with the context node. For instance, the ranking function may favor the results that are topologically close to the context node.

6.1 Problem definition

There are a number of use cases where social search may be helpful. For instance, a user may be searching on a social networking site for another user who she remembers only by the first name. There might be potentially thousands of matching people, but friends-of-friends should be shown first. Another application is a user searching for books, music or movies of a certain genre: items favorited by her friends or friends-of-friends are more likely to be interesting for her. Yet another application is context-aware search, where a user is reading a page with a search form on it and enters a query. Pages linked to by the original page (or close by in terms of clicks) should be presented first.

This problem has been considered by Vieira et al. (Vieira et al., 2007). It can be further formalized as follows: Given a graph $G(V, E)$, a source vertex $q \in V$ and a set $X = \{x_1, x_2, \dots, x_{|X|}\} \subseteq V$ that satisfies some query introduced by node q , compute a permutation $\bar{\pi} = \langle \pi_1, \pi_2, \dots, \pi_{|X|} \rangle$ of the items in the set $\{1, 2, \dots, |X|\}$, so that for the ranking $\langle x_{\pi_1}, x_{\pi_2}, \dots, x_{\pi_{|X|}} \rangle$ it is true that $\nexists \pi_i, \pi_j | i < j \wedge d_G(q, x_{\pi_i}) > d_G(q, x_{\pi_j})$, where $d_G(x, y)$ is the graph-distance of vertices x, y .

In practice, a query is a tuple $\langle q, t \rangle$ where q is the asking vertex and t is a set of keywords or tags that must be matched against all the elements of the collection to select the candidate set X . Note that both the source vertex q and the set of relevant items X are only known at runtime.

6.2 Evaluation method

To evaluate the effectiveness of our methods for search, we need a method for generating searches, as well as a performance metric. For the latter we use *precision@k* denoted by $P@k$; this is the size of the intersection between the top k elements returned by the proposed methods using approximate distances, and the true top k elements in the result set X , normalized by k . Given that there might be elements of X tied by distance, we extend X to include the elements at the same distance as the k -th element on X , as any of those elements can be considered a good result to the query.

It should be pointed out that while this evaluation method emulates a hypothetical ranking function that uses only the distances, in most practical settings the distance between two items should be a component of the ranking function, not a replacement for it.

To generate queries and select the matching results we consider that each element in the graph has a set of *tags* or keywords associated to it. In the case of the Flickr datasets (both explicit and implicit graphs) tags are naturally provided by users and we consider that a user has a tag if she has tagged at least one photo with that tag, filtering out tags that have been used in less than 100 photos. In the case of the Yahoo! Instant Messaging graph, we cross the information with the items users have rated in Yahoo! Movies (we have been provided with an anonymized dataset in which this information is already joined), so the tags of a user are the movies she has rated.

In the case of the Wikipedia dataset, the tags are the words the pages contain. Words are selected that are neither uncommon nor trivially common: 650 words that have frequency in the ranges $[1000, 1050] \cup [2000, 2050] \cup [5000, 5050] \cup [10000, 10050]$. In the case of DBLP, there were no tags available. Thus for experimentation purposes tags were uniformly at random assigned to users; 201 tags were created and each one was assigned to 100 random users in the graph.

Table 6.1: Summary of precision at 5 across datasets, using 20 (top) and 100 landmarks (bottom)

20 landmarks	Fl.-E	Fl.-I	Wiki	DBLP	Y!IM
RANDOM	0.84	0.82	0.42	0.60	0.29
DEGREE	0.98	0.96	0.46	0.77	0.23
CENTRAL	0.98	0.97	0.58	0.77	0.43
DEGREE/1	0.97	0.95	0.46	0.78	0.23
CENTRAL/1	0.97	0.94	0.58	0.80	0.46
DEGREE/P	0.97	0.90	0.50	0.78	0.45
CENTRAL/P	0.97	0.92	0.57	0.78	0.46
BORDER/P	0.98	0.92	0.53	0.79	0.46
100 landmarks	Fl.-E	Fl.-I	Wiki	DBLP	Y!IM
RANDOM	0.86	0.80	0.59	0.65	0.39
DEGREE	0.99	0.98	0.67	0.88	0.25
CENTRAL	0.99	0.98	0.69	0.87	0.51
DEGREE/1	0.98	0.97	0.67	0.92	0.56
CENTRAL/1	0.98	0.96	0.68	0.89	0.58
DEGREE/P	0.98	0.90	0.65	0.88	0.65
CENTRAL/P	0.97	0.95	0.71	0.89	0.64
BORDER	0.99	0.95	0.68	0.91	0.69

6.3 Experimental results

Table 6.1 summarizes the precision at 5 of the heuristics. Again, we are selecting a landmark set of size 20 and 100. The RANDOM technique is clearly outperformed in all datasets.

Figure 6-1 shows how $p@5$ scales using more landmarks, while Figure 6-2 shows the same for $p@10$. The x -axis is logarithmic so it is clear that returns are diminishing; in any case a few tens of landmarks are enough for Flickr and a few hundred landmarks for the other datasets. Table 6.1 is quite consistent with Table 5.2 and similar conclusions hold. Among the simple strategies CENTRALITY is better and DEGREE performs poorly in the Y!IM dataset. The constrained CENTRAL/1 and the partitioning-based BORDER/P yield very good results across all datasets.

As a general observation, the number of landmarks necessary for a good approximation depends much more on the graph structure than on the graph size. For instance, despite Y!IM being the smallest graph in our experiments, the search application seems to require more landmarks to obtain a high precision than for the other datasets. It should be noted that Y!IM graph is a result of

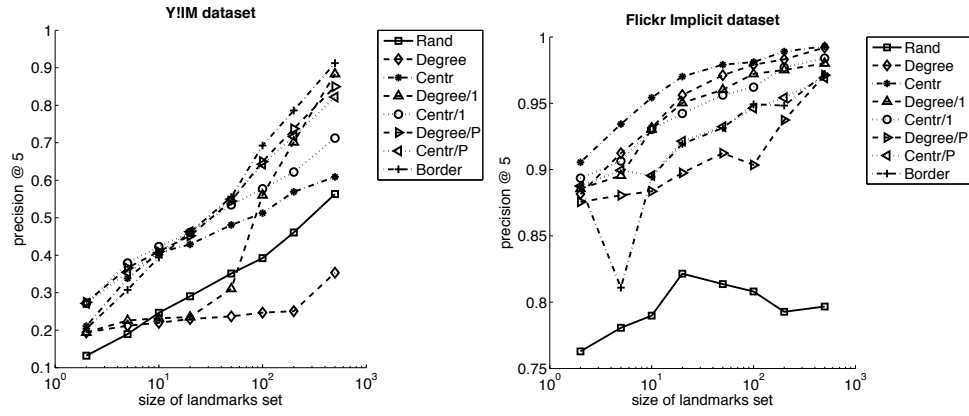


Figure 6-1: Precision at 5 for the social search task

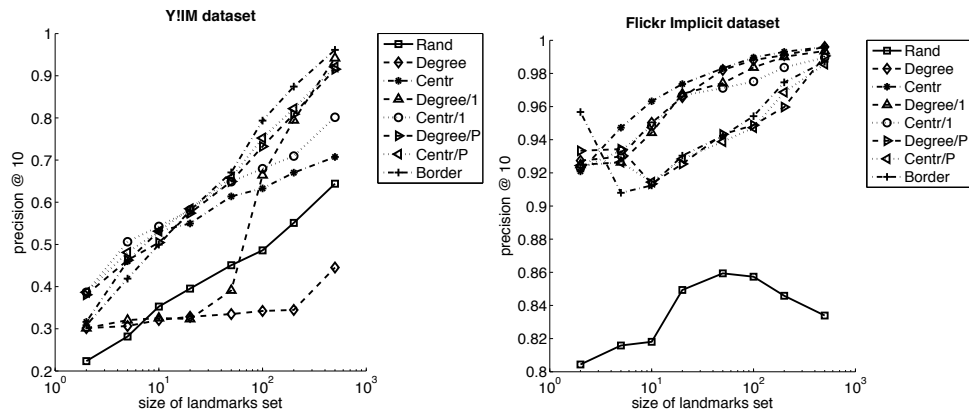


Figure 6-2: Precision at 10 for the social search task

intersecting Y! Movies Ratings with Y! Instant Messaging. Thus it is a sparser graph than the other four, with many single-path parts, that are hard to cover with a small number of landmarks.

Chapter 7

Application to Community Detection

A preliminary investigation on the applicability of landmark-based embeddings to the task of clustering nodes of a graph is presented. The underlying intuition is that, if landmarks are chosen appropriately, node embeddings may contain rich information about the node position within the graph. By using the embeddings as the objects features, and adopting a standard Euclidean distance between embeddings, the community detection task may be reduced to a standard clustering on relational data. The above intuition is put in practice on the DBLP dataset as follows.

Using the DEGREE/1 method, a set of 100 landmarks and the corresponding node embeddings have been generated. Standard K -means (with $K = 50$) is then applied using node embeddings as features.

In order to assess the meaningfulness of the obtained clusters in a qualitative manner the area of publication of the clusters is inspected. In particular *salient journals* inside each cluster are computed as follows. For each journal paper $1/k$ points are given to each of its k co-authors. For each cluster and for each journal the sum of the points is computed for the given journal over all authors in the cluster.

A journal is represented in a cluster if it has more than 20% of its total points associated with the cluster of authors, i.e. the cluster of authors contributes to more than 20% of a given journal history of publications. In Table 7.1 the journals that get more points inside each of those clusters are listed. Note that some clusters may disappear as they may not reach the 20% threshold for any journal.

It is worth noting how two different communities of the large Database community arise: one mixed with artificial intelligence (Cluster 35), and one mixed with data mining and algorithms (Cluster 10).

Table 7.1: Clusters obtained in DBLP.

Cluster 1 - 8693 authors - Graphs, Combinatorics, Discr. Math.	
Contrib. %	Journal
0.32	Algorithmic Operations Research
0.25	Graphs and Combinatorics
0.23	Journal of Graph Theory
0.22	Discrete Applied Mathematics
0.22	Combinatorics, Probability & Computing
0.21	Discrete Mathematics
0.21	Journal of Discrete Algorithms
0.21	Electronic Colloquium on Computational Complexity

Cluster 10 - 5375 authors - DB, KDD, Algorithms	
Contrib. %	Journal
0.67	SIGMOD Digital Review
0.38	IEEE Data(base) Engineering Bulletin
0.35	ACM Transactions on KDD (TKDD)
0.33	Internet Mathematics
0.33	Theory of Computing
0.33	SIGMOD Record
0.32	The VLDB Journal
0.31	Advances in Computing Research
0.31	Algorithmica
0.30	ACM Transactions on Database Systems (TODS)
0.30	Data Mining and Knowledge Discovery
0.30	ACM Transactions on Algorithms

Cluster 16 - 6012 authors - Imaging, Vision, Virtual Reality	
Contrib. %	Journal
0.25	Journal of Mathematical Imaging and Vision
0.22	International Journal of Virtual Reality

Cluster 21 - 5970 authors - Algorithms, Complexity	
Contrib. %	Journal
0.36	Computational Geometry: Theory and Applications
0.34	Theory of Computing
0.33	ACM Transactions on Algorithms
0.31	Journal of Graph Algorithms and Applications
0.31	Random Structures and Algorithms
0.30	Electronic Colloquium on Computational Complexity
0.30	Journal of Automata, Languages and Combinatorics
0.30	Computational Complexity
0.29	Combinatorics, Probability & Computing
0.28	Combinatorica
0.27	Discrete & Computational Geometry
0.27	ACM Journal of Experimental Algorithms (JEA)
0.27	SIAM Journal on Computing
0.27	Journal of Algorithms
0.26	Algorithmica
0.26	Internet Mathematics

Cluster 35 - 6938 authors - DB, Artificial Intelligence	
Contrib. %	Journal
0.34	Journal of Web Semantics
0.27	SIGMOD Record
0.26	The VLDB Journal
0.24	Autonomous Agents and Multi-Agent Systems
0.23	IEEE Data(base) Engineering Bulletin
0.23	ACM Transactions on the Web
0.23	AI Magazine
0.23	Journal of Artificial Intelligence Research (JAIR)
0.22	SIGMOD Digital Review
0.21	Intelligence
0.21	ACM Transactions on Database Systems (TODS)
0.21	Artificial Intelligence

Cluster 44 - 8230 authors - Architecture, Parallelism, Concurrency	
Contrib. %	Journal
0.35	Journal of Instruction-Level Parallelism
0.34	ACM Trans. on Arch. and Code Opt. (TACO)
0.32	Computer Architecture Letters
0.24	ACM Letters on Progr. Lang. and Syst. (LOPLAS)
0.22	International Journal of Parallel Programming
0.22	ACM Transactions in Embedded Computing Systems
0.22	Computer Communication Review (ACM SIGCOMM)
0.21	ACM Transactions on Computer Systems (TOCS)
0.21	IEEE Concurrency
0.20	SIGARCH Computer Architecture News

The results are promising: recall that the nodes have been clustered according to the graph nodes (i.e., the authors), not the journals. Moreover, the clusters are obtained using only the embeddings and no additional information.

Chapter 8

Conclusions and future work

Motivated by applications such as context-aware web search, and socially sensitive search in social networks, this thesis studies how to do fast and accurate distance estimation on real-world very large graphs. In particular, the focus is on approximate methods based on landmarks. The problem of optimal landmark selection has been defined and proven to be **NP**-hard. Thus, several heuristics for landmark selection that outperform the commonly used **RANDOM** by a large margin are defined. In the simplest class of heuristics, **CENTRALITY** appears to be much more robust than **DEGREE**. Among the more elaborate heuristics, the ones based on partitioning, in particular **BORDER/P** are the most promising. When applied to the task of context-aware search in four social networks and a webgraph, these methods result to savings of a factor up to 250 in terms of space and computation time. Finally, the precomputed embeddings may provide a meaningful graph partitioning using standard clustering methods for relational data. A preliminary experimentation on the DBLP coauthors graph detected communities of authors in terms of their research interests.

In future work we consider we intend to investigate methods for an effective synergy of upper and lower bounds. We also plan to investigate which machine learning methods can be efficiently applied to large graphs for the problem of selecting a good set of landmarks and if they yield improvement over the heuristics. Of most interest is to investigate the means to provide estimates for more distance-functions in graphs, which could be used as primitives in context and/or social aware search tasks.

References

- Abraham, I., Bartal, Y., Hubert, T., Kedar, C., Gupta, D. A., Kleinberg, J., Aleks, O. N., and Slivkins, R. (2005). Metric embeddings with relaxed guarantees. In *Proc. 46th IEEE Symp. on Foundations of Computer Science*, pages 83–100.
- Amer-Yahia, S., Benedikt, M., Lakshmanan, L. V., and Stoyanovic, J. (2008). Efficient network-aware search in collaborative tagging sites. In *VLDB '08: Proceedings of the 34th international conference on Very large data bases*.
- Athitsos, V., Papapetrou, P., Potamias, M., Kollios, G., and Gunopulos, D. (2008). Approximate embedding-based subsequence matching of time series. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 365–378, New York, NY, USA. ACM.
- Backstrom, L., Huttenlocher, D. P., Kleinberg, J. M., and Lan, X. (2006). Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'06)*.
- Bader, D., Kintali, S., Madduri, K., and Mihail, M. (2007). Approximating betweenness centrality. pages 124–137.
- Baeza and Ribeiro (1999). *Modern Information Retrieval*. ACM Press / Addison-Wesley.
- Bourgain, J. (1985). On lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52(1):46–52.
- Brandes, U. (2001). A faster algorithm for betweenness centrality.
- Chvatal, V. (1979). A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to Algorithms, Second Edition*. The MIT Press.
- Dabek, F., Cox, R., Kaashoek, F., and Morris, R. (2004). Vivaldi: a decentralized network coordinate system. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, volume 34, pages 15–26, New York, NY, USA. ACM Press.
- Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.
- Floyd, R. W. (1962). Algorithm 97: Shortest path. *Commun. ACM*, 5(6).

- Freeman, L. (1993). On the sociological concept of group: a empirical test of two models. *American Journal of Sociology*, 98:52166.
- Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41.
- Gibson, D., Kleinberg, J. M., and Raghavan, P. (1998). Inferring web communities from link topology. In *Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia (HYPERTEXT'98)*.
- Girvan, M. and Newman, M. (2002). Community structure in social and biological networks. *Proc. Natl. Acad. Sci.*, 99:82718276.
- Goldberg, A. (2007). Point-to-point shortest path algorithms with preprocessing. pages 88–102.
- Goyal, A., Bonchi, F., and Lakshmanan, L. (2008). Discovering leaders from community actions. In *In Proceedings of ACM 17th Conference on Information and Knowledge Management (CIKM 2008)*.
- Hill, B. M. (1975). A simple general approach to inference about the tail of a distribution. *The Annals of Statistics*, 3:1163–1174.
- Hjaltason, G. R. and Samet, H. (2003). Properties of embedding methods for similarity searching in metric spaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(5):530–549.
- Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30.
- Hopcroft, J. E., Khan, O., Kulis, B., and Selman, B. (2003). Natural communities in large linked networks. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'03)*.
- Karypis, G. and Kumar, V. (1995). *MeTis: Unstructured Graph Partitioning and Sparse Matrix Ordering System, Version 2.0*.
- Kleinberg, J., Slivkins, A., and Wexler, T. (2004). Triangulation and embedding using small sets of beacons. In *FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS'04)*, pages 444–453, Washington, DC, USA. IEEE Computer Society.
- Kraft, R., Chang, C. C., Maghoul, F., and Kumar, R. (2006). Searching with context. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 477–486, New York, NY, USA. ACM Press.
- Kumar, R., Novak, J., and Tomkins, A. (2006). Structure and evolution of online social networks. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'06)*.
- Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. (1999). Trawling the web for emerging cyber-communities. *Computer Networks*, 31(11-16):1481–1493.

- Newman, M. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Phys. Rev.*, 69.
- Ng, T. S. E. and Zhang, H. (2002). Predicting internet network distance with coordinates-based approaches. In *INFOCOM*, pages 170–179.
- Singla, P. and Richardson, M. (2008). Yes, there is a correlation: - from social networks to personal behavior on the web. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 655–664, New York, NY, USA. ACM.
- Tang, L. and Crovella, M. (2003). Virtual landmarks for the internet. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 143–152, New York, NY, USA. ACM.
- Thorup, M. and Zwick, U. (2001). Approximate distance oracles. In *ACM Symposium on Theory of Computing*, pages 183–192.
- Ukkonen, A., Castillo, C., Donato, D., and Gionis, A. (2008). Searching the wikipedia with contextual information. In *ACM CIKM*.
- Venkateswaran, J., Lachwani, D., Kahveci, T., and Jermaine, C. (2006). Reference-based indexing of sequence databases. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 906–917. VLDB Endowment.
- Vieira, M. V., Fonseca, B. M., Damazio, R., Golgher, P. B., Davi, and Ribeiro-Neto, B. (2007). Efficient search ranking in social networks. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 563–572, New York, NY, USA. ACM.
- Wasserman, S. and Faust, K. (1994). *Social Network Analysis*. Cambridge University Press, Cambridge, MA.
- Zwick, U. (2001). Exact and approximate distances in graphs — a survey. *Lecture Notes in Computer Science*, 2161.