Orion: Shortest Path Estimation for Large Social Graphs

Xiaohan Zhao, Alessandra Sala, Christo Wilson, Haitao Zheng and Ben Y. Zhao
Department of Computer Science, UC Santa Barbara, USA
$\{$*xiaohanzhao, alessandra, bowlin, htzheng, ravenben*$\}$*@cs.ucsb.edu*

Abstract

Through measurements, researchers continue to produce large social graphs that capture relationships, transactions, and social interactions between users. Efficient analysis of these graphs requires algorithms that scale well with graph size. We examine node distance computation, a critical primitive in graph problems such as computing node separation, centrality computation, mutual friend detection, and community detection. For large million-node social graphs, computing even a single shortest path using traditional breadth-first-search can take several seconds.

In this paper, we propose a novel node distance estimation mechanism that effectively maps nodes in high dimensional graphs to positions in low-dimension Euclidean coordinate spaces, thus allowing constant time node distance computation. We describe Orion, a prototype *graph coordinate system*, and explore critical decisions in its design. Finally, we evaluate the accuracy of Orion's node distance estimates, and show that it can produce accurate results in applications such as node separation, node centrality, and ranked social search.

1 Introduction

Analysis of graph properties is critical to understanding the mechanisms underlying the formation and evolution of complex networks, and is of particular importance in the study of online social networks. In recent years, the research community has seen a rise in large-scale measurement studies of deployed social networks [2, 18] and interaction networks [15, 32], some producing graphs of up to tens of millions of nodes. The size of these massive graphs makes their analysis extremely challenging, as even efficient algorithms can become time-consuming.

Computing node distance, or the shortest-path distance between two nodes, is a primitive that lies at the core of both graph analysis algorithms and social network applications. For example, in a network with n nodes, computing exact values for node separation metrics like graph radius, graph diameter, and average path length, requires calculating $O(n^2)$ node distances. In deployed social networks, LinkedIn users can use node distance to filter out query results in their neighborhood, and social e-commerce sites can use node distance to identify more trustworthy sellers [27]. Node distance is also the determining factor for other common graph problems like centrality and mutual friend detection.

Current methods for computing node distance do not scale with graph size. For a graph with n nodes and m edges, efficient implementations of traditional algorithms including breadth-first-search (BFS), Dijkstra and Floyd-Warshall can produce shortest paths for each node pair in $O(n \log n + m)$ time, and all pairs shortest-paths in $\Theta(n^3)$ [6]. Tolerable for small graphs, the computation required for a single node distance computation on a large million-node graph can take up to a minute on modern computers [23]. Given the prohibitively high costs of storing precomputed distances, researchers have little choice but to sample portions of the graph or seek approximate results.

In this paper, we propose a novel approach to approximating node distance measurements we call *Graph Coordinate Systems*. A graph coordinate system maps nodes in high dimensional graphs to positions in a fixed-dimension Euclidean coordinate space. Using the coordinates associated with each graph node, we can use a simple Euclidean distance computation to estimate, in constant time, its distance to any other node in the graph. Our work is inspired by the prior success of using virtual network coordinate systems [7, 8, 20] to predict latencies between Internet hosts. Studies show that integrating network coordinates into applications such as web caches and peer-to-peer systems significantly improved their performance. Unlike latencies between Internet hosts, however, shortest path values on a graph, by definition, will never violate the triangle inequality [17]. Since triangle inequality violations are often cited as a key source of error in network coordinate systems, graph coordinates could potentially be even more accurate.

We make three key contributions in this paper. First, we propose the use of graph coordinate systems to simplify node distance computation on large graphs. While similar in fundamental methodology to network coordinates, several critical differences force a ground-up redesign of graph coordinate systems. For example, while network coordinates can be easily tuned using fast latency measurements (*e.g.* via Internet ping), measuring actual distances between graph nodes can be very expensive. We describe Orion, a prototype graph coordinate system, and explore critical decisions in its design. Second, we perform extensive validation of Orion's node distance estimates using several real social graphs. Finally, we explore the utility of graph coordinate systems in graph analysis and social applications, and show that Orion produces effective results on large graphs for applications such as node separation metrics, centrality computation, and ranked social search.

Roadmap. We begin in Section 2 by defining our goals and assumptions, and describing key differences from prior work on network coordinate systems. We then describe the Orion graph coordinate system and explain key design decisions in Section 3. Next, we present accuracy measurements of Orion in Section 4, and show the effectiveness of Orion in computing graph metrics and graph applications in Section 5. Finally, we discuss future directions and conclude in Section 6.

2 Virtual Coordinates and Large Graphs

The goal of our work is to find a compact representation of distances between nodes in a graph, such that we can quickly and easily compute estimates of shortest path distances between any two nodes. We are inspired by the significant volume of prior work on the topic of network coordinate systems, much of which mapped distances between Internet hosts to distances in a Euclidean space. In this section, we briefly summarize prior work in network coordinates, and use it as context to identify key differences and challenges in the design of graph coordinate systems. Finally, we briefly discuss related projects as context for our work.

2.1 Background: Network Coordinates

Network coordinate (NC) systems [7, 8, 17, 20, 21, 22, 29] were designed as efficient and scalable mechanisms to estimated distances or latencies between Internet hosts. Such distance estimation mechanisms can prove critical to large-scale distributed systems that use approximate distance values for performance optimization. Applications that benefit from these systems include content distribution networks [24], multicast systems [3], distributed file systems [26] and file-sharing networks [1, 5].

The majority of network coordinate systems work by mapping an Internet host to a specific position in a Euclidean space based on round-trip measurements to other hosts. Depending on the protocol, a node's coordinates can be continually refined as additional measurement results are added to the system. Once a pair of nodes has converged to their positions in the coordinate space, their distance in the Internet (usually a round-trip-time or RTT value) can be predicted by computing the Euclidean distance between their coordinate values.

Based on the way coordinates are computed for new nodes, NC systems can be generally categorized into "landmark-based" and "decentralized" systems. Landmark-based systems such as GNP [20] first compute coordinates for an initial set of well-known landmark nodes using pair-wise measurements, where errors between virtual and measured distances are minimized using a non-linear optimization algorithm such as Simplex Downhill [19]. The NC then uses these nodes as fixed points to calibrate coordinate values for the rest of the network. Landmark-based systems [17, 20, 21, 22, 29] have fast convergence properties, since all nodes rely on the same fixed nodes for their coordinate calculations. However, the accuracy of these systems can suffer if the choice of landmark nodes is suboptimal, *i.e.* they do not sufficiently cover the network.

In contrast, decentralized NCs such as PIC [7] and Vivaldi [8] allow incoming nodes to orient themselves in the coordinate space using any nodes already positioned in the space. While these systems avoid dependence on well-known landmarks, new nodes can force already calibrated nodes to adjust their coordinates, potentially increasing convergence time and propagating errors. For further details on NC systems, we refer the reader to a recent survey [9].

Successes and Limitations. NC systems have been shown to be highly effective at improving performance of large distributed systems [12, 1]. However, more recent work has questioned the validity of using Euclidean spaces to approximate Internet latencies, which have been shown to violate the Triangle Inequality [13, 33].

2.2 Graph Coordinates: Challenges

Our goal is to investigate the feasibility of using a Euclidean coordinate space to capture node distances on large graphs. Upon consideration, we find that three key differences separate the problems of estimating shortest paths on graphs and host latencies on the Internet. As a result, we cannot simply apply techniques from NC systems, but must instead carefully reevaluate them in the context of graph distances.

Triangle Inequality. First, we note that while the presence of triangle inequality violations (TIV) is often identified as a barrier to accuracy in network coordinate systems, shortest path computation on graphs is guaranteed to be TIV free. This is inherent in the definition of the shortest path metric. The proof is straightforward by contradiction. Assume a triangle inequality violation for three nodes a, b, c, *i.e.* $d(a, b) + d(a, c) < d(b, c)$, where $d(a, b)$ represents the shortest path distance between nodes a and b. This scenario is impossible, because one can construct a "shorter" shortest path between b and c that is the concatenation of the shortest paths between (b, a) and (a, c). At minimum, the sum of lengths of two shortest paths in the triangle is equal to the length of the third. This property means a graph coordinate system does not have to support TIVs by resorting to complex algorithms such as matrix factorization [17].

Cost of Measurements. The second and most critical difference between these two problems is the cost of obtaining ground truth distance values between two nodes. In Internet latency estimation, a running system can perform a latency measurement with minimal cost via Internet Ping. In contrast, measuring the shortest path between graph nodes is expensive, and can take at worst time $O(n+m)$. In addition, computing the distance from a to b using BFS effectively computes the shortest path between a to all other nodes in the graph. With these factors in mind, we must carefully consider how graph coordinates obtain real node distances for node calibration. We must minimize the number of overall BFS operations, while reusing the results from each BFS operation as much as possible.

Error Sensitivity. Finally, graph coordinate systems face an additional challenge of higher error sensitivity. While latency between Internet nodes can vary from submilliseconds to hundreds of milliseconds, node distances on small-world graphs tend to have much smaller variance. For example, diameters of recently measured Facebook graphs are less than 20 [32]. Additionally, all node distance values are integers. This means node distance values across different paths in a graph are significantly more clustered across a small number of possible values, and any estimation errors can be rounded up. Thus, a graph coordinate system must provide reasonably high accuracy in order to be useful in graph applications.

2.3 Related work

Shortest Path Methods. Shortest path computations are extremely costly on large graphs. Rattigan et al. proposes to compute nodes position in a graph by exploiting a coordinate-like approach, called network structure index (NSI) [25]. Compared to Orion, NSI is more expensive in both time and space complexity. The space complexity of NSI is $O(nkD)$, where k is the number of zones and D is the number of dimensions, which are k times higher than Orion. On the other hand, NSI's time complexity, $O(mkD)$, is proportional to the number of edges m while Orion takes only $O(nkD)$ time, where n is the number of nodes. This also represents a significant decrease in time complexity, since m is several orders of magnitude larger than n in online social graphs. Furthermore, unlike our work, annotation distances computed by NSI are not the number of hops between nodes pairs.

Recent work by Potamias et al. [23] proposes a landmark scheme for approximating shortest path distances. The approach is similar in spirit, but stores for each node its distance to every landmark. In contrast, Orion is more compact. It stores for each node a coordinate address of *e.g.* 10 values, independent of the number of landmarks used. In addition, our work considers the broader problem of embedding large graphs into known coordinate spaces, and evaluates our work using a broad array of applications.

Social Networks. A significant amount of research effort has been invested to understand OSNs such as MySpace, Orkut [2], Flickr, LiveJournal [18], Facebook [32], and Twitter [10]. Social networks are characterized by graph properties like power-law degree distribution, small-world clustering, and scale-free behavior [16]. A necessary precondition for quantifying some of these characteristics is calculating node separation metrics (*i.e.* radius, diameter and average path length) that are based on all-pairs shortest paths. Some social applications also leverage shortest path computations, such as distance-based community detection [11]. Unfortunately, computing all-pairs shortest paths on today's social graphs is infeasible, since they often have millions of nodes and hundreds of millions of edges. Existing studies sidestep this issue by using sampling techniques to estimate the graph's true values [18, 32]. In contrast, our solution computes shortest paths between node pairs in 0.2 microseconds, making it a scalable solution for computing all-pairs shortest paths on massive social graphs.

3 Designing Orion

In this section, we present the Orion graph coordinate system and explain our design decisions in detail. Similar to network coordinate systems, graph coordinate systems work in two phases. First, nodes in the graph are iteratively added to the coordinate space, the position of each node being calibrated by ground truth

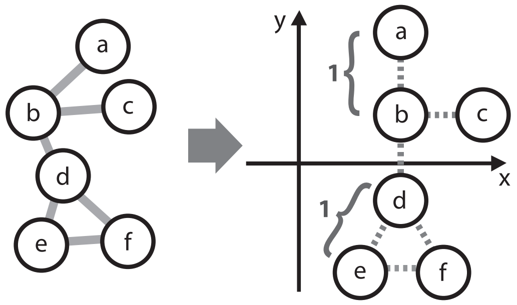

Figure 1: Mapping graph nodes into Euclidean coordinate space. For most node pairs, the Euclidean distance exactly matches the hop-count separating them in the original graph.

node-distance measurements. This “calibration phase” is where a graph coordinate system incurs its one-time computational overhead. Once all nodes in the graph have been added, the resulting system can be integrated with graph applications to answer node distance queries with estimates.

Since the per-query computation cost is $O(1)$, the focus of our design is to ensure the calibration phase is computationally efficient, and the results are as accurate as possible. More specifically, our goals are three-fold:

- Scalability. The computational cost of the calibration phase must scale linearly with the number of nodes, *i.e.* $O(n)$.
- Accuracy. While individual node distance predictions might incur reasonable errors, predictions should approximate ground truth at the large scale.
- Fast convergence. Impact of individual node calibrations should be localized, *i.e.* should not trigger significant new adjustments to their neighbors.

Based on these goals, we now describe the Orion design and explain key decisions.

3.1 A Landmark-based Approach

Figure 1 illustrates how Orion maps nodes in a graph to positions in a D-dimension Euclidean coordinate space. The goal is accurately translate pairwise hop-count distances in the graph into Euclidean distances in the coordinate space. To do this, Orion uses a landmark approach, where the positions of all nodes are calibrated with their relative distances to a fixed number (k) of chosen landmark nodes. Landmark nodes are initially chosen from the entire graph based on their position and degree of connectivity.

Why Landmarks? We use a landmark-based scheme in Orion for two main reasons. First and foremost, we wish to minimize the number of shortest path computations needed to establish ground truth on the actual graph, since each computation can, in the worst case, require a full traversal of the graph. Using a landmark approach, we limit the total number of Breadth-First-Search operations to k, the number of landmarks. Each BFS computes the shortest path distance from a landmark to all other nodes. Computing BFS for all landmarks essentially precomputes all values needed to calibrate all nodes in the graph. In contrast, a decentralized approach such as the physical springs model used by Vivaldi [8] requires shortest path computations between random node pairs, thus drastically increasing the number of BFS operations.

The second advantage of a landmark-based scheme is that the positions of incoming nodes depend only on the landmark nodes. This bounds the number of operations required to compute a node's position, guaranteeing fast convergence. In contrast, in decentralized models adding a new node will often force its nearby neighbors to make adjustments on their position, a process that can propagate adjustments iteratively throughout the entire space.

Finally, we note that the challenges that make Landmark systems undesirable in Internet systems do not apply in our context. In network coordinate systems, landmarks are physical machines that must remain available at all times, and processing load from other applications (*e.g.* web traffic) can affect the accuracy of latency measurements to other machines in the network [21]. Compromised landmarks can also significantly impact the entire system [9]. Those issues do not exist for graph coordinates, where nodes are just graph vertices and all computation can be performed on a centralized server.

3.2 Scalable Landmark Coordinates

Intuitively, the number of landmarks used to calibrate a graph should have a direct impact on the accuracy of the Euclidean mapping. Similar correlation between landmarks and accuracy has been observed in the context of network coordinate systems [20]. The highly connected and complex nature of social graphs leads us to believe that an accurate graph coordinate system requires a significant number of landmarks. The challenge is to find a way to accurately and quickly compute the coordinates for a large number of landmarks.

Traditional network coordinates determine a node's D-dimension coordinates by minimizing the sum of squares of prediction errors using the Simplex Downhill algorithm [19], a nonlinear optimization algorithm. The algorithm runs in $O(k^2 \cdot D)$ time to compute coordinates of k landmarks.

Since running Simplex Downhill on our desired number of landmarks (up to 100 in our study) is computationally expensive, we propose a new approach, where we separate our landmarks into two groups, a small initial group of 16 landmarks, and a larger secondary group composed of the remaining landmarks.

We leverage the Simplex Downhill algorithm to compute the coordinates for the initial ($k_I = 16$) landmarks, thus its asymptotical complexity is $O({k_I}^2 \cdot D)$. The secondary group of landmarks calibrate their positions using the initial k_I landmarks as anchors, contributing to a computational complexity of only $O(k_I \cdot D)$ each. Thus, the total time required to compute landmark coordinates is $O({k_I}^2 \cdot D) + (k - k_I) \times O(k_I \cdot D)$, where k is the total number of landmarks.

Furthermore, we describe two ways to compute the coordinates of the secondary group of landmarks, while maintaining the same computational complexity. In the *global approach*, we compute the coordinates of each node in the secondary group relying only on the initial group as anchors. In the *incremental landmarks approach*, nodes in the secondary group are added one by one. Once a node receives its coordinate values, it becomes an anchor for all remaining nodes. To compute its coordinates, any remaining node in the secondary group can choose any k_I nodes from all embedded nodes to be its landmarks.

3.3 Landmark Selection

Finally, we consider the problem of choosing landmark nodes to produce the most accurate graph to Euclidean coordinate mapping. Prior work by Potamias et. al considered the problem of choosing landmarks, and concluded experimentally that choosing nodes with high centrality performed significantly better than random choice [23]. Given the complexity of computing node centrality, we consider two groups of alternative landmark selection strategies as possible approximations of centrality-based selection: Random and High-degree.

- *Random.* This is the basic landmark selection strategy. Landmarks are chosen uniformly at random from all nodes in the graph.
- *High-degree.* Prior measurements on social networks [18, 32] show that social graphs exhibit a power-law-like degree distribution. Intuitively, high degree nodes reside at the core of social graphs, effectively approximating central nodes. This strategy chooses nodes with the highest degree.
- *Landmark separation.* Closely positioned landmarks are less effective at “covering” the graph as anchors. Therefore, we add variants to the two basic strategies, where we select the landmarks one by one, ignore any potential landmarks that are too close in the graph to existing landmarks, and continue selecting landmarks until the desired number has been met.

Network	Nodes	Edges	Avg. Path Len.
Norway	293K	5,589K	4.2
Egypt	246K	1,618K	5.0
Los Angeles	275K	2,115K	5.1
India	363K	1,556K	6.1

Table 1: Properties of Social Graphs

We consider these strategies as approximations of the high-centrality strategy, and evaluate their effectiveness empirically in Section 4.

Summary. Orion works as a landmark-based scheme, where an initial core of 16 landmarks is first fixed in the space using Simplex Downhill optimization. A secondary group of landmarks position themselves based on the original landmarks. Finally, all remaining graph nodes calibrate their positions based on node distances obtained from computing BFS from all landmarks.

4 Experimental Results

In this section we analyze the accuracy of Orion’s node distance estimates. We study the impact on accuracy by key factors: Landmark selection strategy, cardinality of the Landmark set, and dimensionality of node coordinates. We preface our core discussion with an overview of the experimental environment and evaluation metrics.

4.1 Experimental Setup

We evaluate Orion accuracy using four anonymized datasets (Egypt, India, Los Angeles and Norway) gathered from Facebook regional networks [32]. These graphs were chosen because they are large, but not too large to make graph analysis intractable. Their statistical properties are consistent with other OSN datasets [2, 30]. Table 1 reports their basic properties.

All experiments were run on 2.4 GHz, dual core Xeon servers with 32GB of RAM. All machines ran Fedora Core, kernel version 2.6.x.

Evaluation Metrics. We use two key metrics to evaluate Orion accuracy. The first is *Relative Error*. This metric is widely used in the study of Network Coordinate Systems, although it must be modified slightly in order to evaluate graph coordinate systems. Let a and b be two nodes in the graph. Let $d^m_{a,b}$ be the *measured* distance between a and b on the real graph using the BFS algorithm, and let $d^P_{a,b}$ be the *estimated* distance computed using a and b’s coordinates from Orion. In our context, the relative error is:

$$Re = \frac{|d^m_{a,b} - d^P_{a,b}|}{d^m_{a,b}} \tag{1}$$

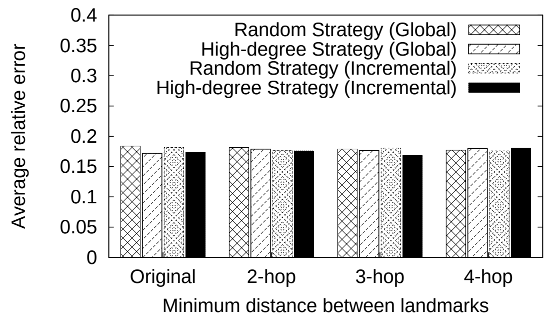

Figure 2: ARE of nodes' distances with different combination of landmark selection and computation strategies in India graph

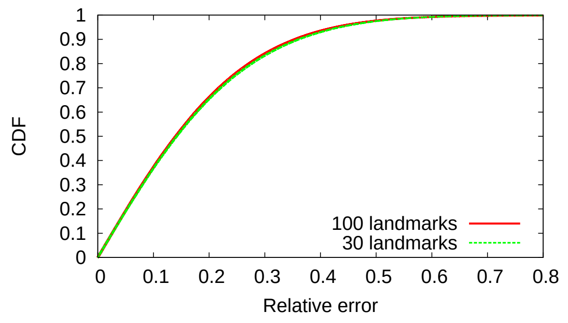

Figure 3: CDF of relative error on nodes distances on India.

The second metric is *Average Relative Error* (ARE) of predicted distances. Small ARE values are sufficient to prove that the majority of node pairs in Orion have realistic predicted distances. Finally, we also use *Computation time* to investigate Orion's efficiency.

4.2 Estimation Accuracy

We examine Orion's estimation accuracy under the influence of three different factors: landmark selection strategy, cardinality of the landmark set, and dimensionality of node coordinates.

Landmark Selection Strategies. We begin by analyzing the impact of landmark selection strategies on accuracy. In Section 3.3, we describe two selection strategies (random and high-degree) and variants based on landmark separation. Figure 2 plots AREs for a variety of landmark selection strategies using the India graph. We evaluate the accuracy of each different strategy on all four datasets. These results are similar for all our graphs, and we only show India here for brevity.

Each evaluation is performed by selecting 1000 random nodes in the graph and computing pairwise distances between them, for a total of $\approx 500K$ distances. These results form the control sample when calculating relative error vs. Orion. Each value reported in Figure 2 is the average results over 5 sets of randomly selected 1000 node groups.

In general, Figure 2 shows that Orion provides low relative errors compared to actual path lengths for different landmark selection strategies. Among the considered strategies, Figure 2 shows that high-degree strategies can produce lower errors. Furthermore, the impact of landmark separation on the accuracy of shortest path length estimation is fairly small. Taking a close look, the high-degree incremental landmark selection strategy with 3-hop separation provides the most accurate result among all the considered strategies. As a result, all remaining experiments run with this approach.

Cardinality of Landmark Set. In this section we explore the variation in accuracy when we initialize Orion using different cardinalities of the landmark set. The intuition behind this experiment is that by having more landmarks spread in the graph there is a better space coverage that should allow higher precision while placing nodes into this space.

Figure 3 depicts the cumulative distribution function of the relative error for cardinality 30 and 100 of the landmark set. Figure 3 shows that there is a small increase in precision with larger landmark set sizes. In general, almost 70% of the computed distances have a relative error less then 0.2 and more than 90% are less than 0.4, that allows us to validate a satisfactory accuracy in computing node distances with a relatively small landmarks (i.e. 100 landmarks represent a millesimal of our graphs).

Dimensionality of Coordinates. Nodes are mapped into geometric space based on the coordinates they acquire during the initialization phase. Intuitively, calibrating node positions using a larger coordinate vector should have a direct impact on the precision of the estimated distances between nodes.

We compute coordinates as dimensionality varies between 2 and 14. Figure 4 shows that increasing the coordinates dimension also increases the predicted distances between nodes, confirming our intuition. Although higher dimensions produce smaller errors, as the dimension increases the time for coordinate and distance computation increases as well. We explore the trade off between predicted precision and efficiency and conclude that using 10-dimensional coordinates is best compromise. In particular, as shown in Figure 4, the accuracy gain for $x \geq 10$ slightly decreases.

4.3 Computational Complexity

In this section we investigate Orion efficiency by analyzing Orion bootstrap and pair distance computation time versus BFS.

Orion bootstrap involves two main operations: (i)

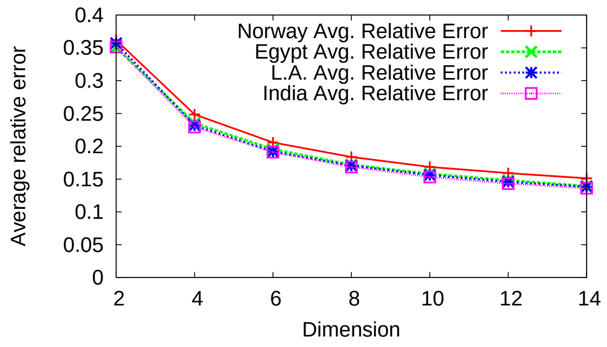

Figure 4: ARE of different coordinate dimensions.

Time	India	Egypt	L. A.	Norway
Orion Bootstrap	9499s	7852s	8856s	9383s
Orion Response	0.2μs	0.2μs	0.18μs	0.19μs
BFS Response	1.028s	0.75s	1.027s	1.44s

Table 2: Computation times for Orion and BFS.

measure distances from each landmark to all the nodes using BFS, and (ii) compute coordinates using Simplex Downhill. We record the time for bootstrapping Orion on our four social graphs and show that Orion bootstrap time is about 2 hours (as shown in Table 2). These times are acceptable since bootstrapping is a one-time cost.

Response time is the average time to compute pairwise node distances using Orion. As shown in Table 2, Orion is 7 orders of magnitude faster than BFS. This result confirms the huge gain a coordinate graph system like Orion is able to achieve compared to traditional methods.

Note that Orion bootstrap and response times are functions of the number of nodes in the graph. Conversely, BFS computation time is a function of the number of edges. Thus Orion is likely to provide better scalability than BFS because, as social networks expand, the growth in edges far surpasses the growth in nodes.

5 Using Orion in Graph Applications

To demonstrate Orion's utility and accuracy in an operational setting, we integrate Orion into several graph analysis and social applications that make extensive use of shortest path computations. Under normal conditions, these graph metrics and applications can be computationally intractable for large graphs. We show that we can use Orion to scalably obtain answers that reasonably approximate answers obtained from deterministic methods. Specifically, we look at three common operations: computing node separation metrics such as graph radius, diameter and average path length, locating central nodes in a graph, and ranked social search.

Metric	Method	India	Egypt	L. A.	Norway
Radius	Orion	11.7	9.5	10.8	8.1
	Actual	11	9	11	8
Diameter	Orion	17.9	13.9	17.8	12.1
	Actual	17	13	17	12
Avg. Path Length	Orion	5.8	4.8	4.9	4.1
	Actual	6.1	5.0	5.1	4.2

Table 3: Comparing Node Separation Metrics for a 1000-node sample in each of our four graphs. Orion's approximations are compared to results computed via BFS.

5.1 Node Separation Metrics

Node separation metrics are commonly used to characterize overall graph structure. The common node separation metrics include *graph radius*, *graph diameter* and *average path length*. The *eccentricity* of a node is defined as the longest hop distance from it to all other nodes in a graph. Graph radius is defined as the minimum eccentricity across all nodes, while graph diameter is defined as the maximum eccentricity across all nodes. Average path length is the mean of all shortest path lengths.

Computation Time. Given their intensive use of shortest path computations, node separation metrics are an ideal application for Orion. We would like to quantify Orion's accuracy in this context by computing these metrics using Orion and compare them directly to those from BFS. Given the large sizes of our graphs, however, it was not possible for us to compute eccentricity for all the nodes by BFS for direct comparison. From our time measurements of single node full BFS we estimate a full computation of the Los Angeles network (275K nodes) would take roughly 152 hours or more than 6 days of computation. In contrast, embedding the LA network into Orion takes less than 2 hours, and querying for all pairwise paths takes roughly 7000 seconds, for a total process time of less than 4 hours.

Accuracy Results. For a scalable side-by-side comparison, we randomly sample 1000 nodes from each of the graphs, and compute graph radius, diameter and average path length based on BFS from those nodes to all other nodes in the graph. We compare those results to those generated using node distance estimations from Orion, and show the results in Table 3. We find that Orion performs very well in predicting these metrics. For graph radius and diameter, it always provides a result that is less than 1 hop from the BFS answer. In the case of average path length, Orion is even more accurate, and provides results that never deviate more than 0.3 from BFS.

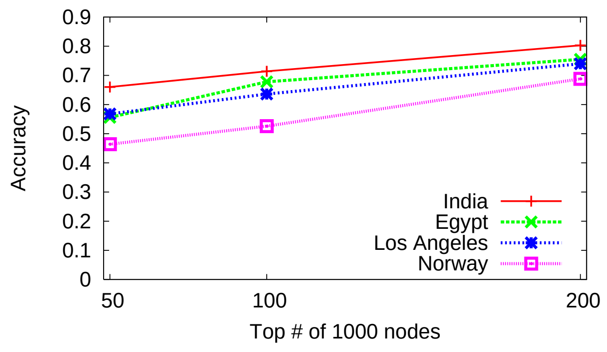

Figure 5: Accuracy of Top k high centrality nodes

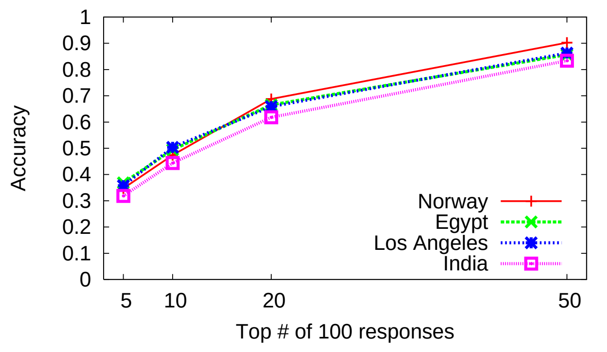

Figure 6: Accuracy of top k ranked nodes.

5.2 Computing Node Centrality

Information dissemination is an active research area of social networks. Viral spread [31], influence campaigns [4, 14], and breaking-news coverage [10] are all examples of information dissemination problems on social graphs. A critical, but computationally expensive, metric necessary for these applications is *node centrality*. We leverage Orion coordinates to compute node's centrality in order to compare its speed and accuracy with centrality calculations performed using traditional shortest-path algorithms.

Centrality is defined as the average shortest path length from a node a to every other nodes in the graph. The smaller the average path length for a node is, the higher its centrality is. Using Orion, a node can quickly estimate its centrality by computing its average Euclidean distance to all other nodes in the graph.

We estimate the precision of computing node centrality via Orion by comparing its results to actual results computed using BFS. Computationally, node centrality also requires all pairs of shortest paths computation, and our time estimates from node separation metrics also apply here (152 hours for our LA graph).

Accuracy Results. To keep computation time manageable, we again sample 1000 random nodes from each graph, and compute node centrality values for each node using both Orion and BFS. We sort nodes based on their average shortest path length to every other node in the network, in increasing order. Then we select the top k nodes from each resulting group, and count the number of top k central nodes (according to BFS) that also appeared in Orion's results. We repeat this for 5 sets of 1000 random nodes and average the result.

Figure 5 shows the percentage of top-k nodes that are correctly considered found by Orion, for different values of k: 50, 100, and 200. The overlap between Orion and BFS' results increases with k. As with results in Section 4.3, centrality results for India are more accurate because it has the longest average path lengths of our sample graphs. The results are generally good across the board, with Orion giving correct estimates more than 50% of the time, when selecting top 50 highest centrality nodes out of 1000.

5.3 Ranked Social Search

Online social networks often need to rank their query results by proximity in the social graph to the query owner. For example, searches for specific names on Facebook and LinkedIn will only return the top results that are closest in social distance to the user. Social distance is used to rank query results because users generally care about people close to their social circles.

We implement a ranked social search application. In each graph, we randomly select 100 nodes to represent the total set of results for each query. We run the simulation 5000 times, each time with a randomly chosen node as the point of origin for the query.

Accuracy Results. We sort the randomly selected 100 nodes in increasing order and choose the top k nodes. Then we count the amount of overlap in the two sets of top k nodes computed by Orion and the BFS-based approach. We define the accuracy of the ranked social search in Orion as the ratio of the number of overlapping nodes to the total number of all considered nodes. Figure 6 plots the accuracy values over different values of k, averaged across the 5000 runs. Again, Orion's social search produces fairly good results, with more than 60% overlap when choosing the top 20 responses.

6 Conclusions and Future Directions

Shortest path computation is one of the most critical and computationally intensive primitives for both graph analysis and social networking applications. We propose *graph coordinate systems*, a new approach to dramatically reduce the complexity of shortest paths computation by mapping the entire graph into a multi-dimensional Euclidean coordinate space. We describe the design of Orion, an efficient graph coordinate prototype. Mapping a graph of n nodes takes time $O(k_I \cdot D \cdot n)$ (roughly 2-3 hours for a 275K node graph), after which each node distance estimation takes less than 0.2 microseconds. Our experiments show Orion can provide accurate results both for graph metrics such as graph radius and node centrality, as well as graph-based applications such as ranked social search.

Future Directions. We believe graph coordinate systems are a promising new research direction for scalable graph analysis. While our work here is preliminary, we see three immediate areas for future work. First, we would like to explore the efficacy of mapping graphs to non-Euclidean coordinate systems such as spherical and hypercube. Second, we will examine the impact of graph coordinates on weighted graphs, *e.g.* geographical graphs or temporal distance metrics for social graphs [28]. Finally, Orion is designed for static graphs. Adding new nodes to the graph after the initial mapping can change shortest path values for portions of the graph and force a re-mapping of the graph. We will investigate mechanisms and heuristics to allow run-time modifications to graphs already mapped to the coordinate space.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under grants IIS-847925, CNS-0916307, and CAREER CNS-0546216.

References

[1] Azureus-vuze. `http://sourceforge.net/projects/azureus/`.

[2] AHN, Y.-Y., ET AL. Analysis of topological characteristics of huge online social networking services. In *Proc of WWW* (2007).

[3] CASTRO, M., ET AL. Scribe: A large-scale and decentralized application-level multicast infrastructure. *IEEE JSAC 20*, 8 (2002).

[4] CHEN, W., WANG, Y., AND YANG, S. Efficient influence maximization in social networks. In *Proc. of ACM KDD* (2009).

[5] COHEN, B. Incentives build robustness in bittorrent. In *Proc. of P2P-Econ* (June 2003).

[6] CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L., AND STEIN, C. *Introduction to Algorithms*, 3 ed. MIT Press, 2009.

[7] COSTA, M., CASTRO, M., ROWSTRON, A., AND KEY, P. Pic: Practical internet coordinates for distance estimation. In *Proc. of ICDCS* (Tokyo, Japan, March 2004).

[8] DABEK, F., COX, R., KAASHOEK, F., AND MORRIS, R. Vivaldi: A decentralized network coordinate system. In *Proc. of SIGCOMM* (Portland, OR, August 2004).

[9] DONNET, B., GUEYE, B., AND KAAFAR, M. A. A survey on network coordinates systems, design, and security. *IEEE Communication Surveys & Tutorials* (2009).

[10] KWAK, H., LEE, C., PARK, H., AND MOON, S. What is twitter, a social network or a news media? In *Proc. of WWW* (2010).

[11] LANCICHINETTI, A., AND FORTUNATO, S. Community detection algorithms: A comparative analysis. *Phys. Rev. E 80*, 5 (Nov 2009).

[12] LEDLIE, J., GARDNER, P., AND SELTZER, M. I. Network coordinates in the wild. In *Proc. of NSDI* (April 2007).

[13] LEE, S., ZHANG, Z., SAHU, S., AND SAHA, D. On suitability of euclidean embedding of internet hosts. In *Proc. of SIGMETRICS* (June 2006).

[14] LESKOVEC, J., ET AL. Cost-effective outbreak detection in networks. In *Proc. of KDD* (2007).

[15] LESKOVEC, J., AND HORVITZ, E. Planetary-scale views on a large instant-messaging network. In *Proc. of WWW* (2008).

[16] LI, L., ET AL. Towards a theory of scale-free graphs: Definition, properties, and implications. *Internet Math 2*, 4 (2005), 431–523.

[17] MAO, Y., SAUL, L., AND SMITH, J. M. Ides: An internet distance estimation service for large networks. *IEEE JSAC 24*, 12 (Dec. 2006), 2273–2284.

[18] MISLOVE, A., ET AL. Measurement and analysis of online social networks. In *Proc. of IMC* (Oct 2007).

[19] NELDER, J. A., AND MEAD, R. A simplex method for function minimization. *The Computer Journal 7*, 4 (Jan. 1965), 308–313.

[20] NG, T. S. E., AND ZHANG, H. Predicting internet network distance with coordinates-based approaches. In *Proc. of INFOCOM* (New York, NY, June 2002).

[21] NG, T. S. E., AND ZHANG, H. A network positioning system for the internet. In *Proc. of USENIX ATC* (June 2004).

[22] PIAS, M., ET AL. Lighthouses for scalable distributed location. In *Proc. of IPTPS* (Feb. 2003).

[23] POTAMIAS, M., ET AL. Fast shortest path distance estimation in large networks. In *Proc. of CIKM* (Hong Kong, Nov. 2009).

[24] RATNASAMY, S., HANDLEY, M., KARP, R., AND SCHENKER, S. Topologically-aware overlay construction and server selection. In *Proc. of INFOCOM* (2002), IEEE.

[25] RATTIGAN, M. J., MAIER, M., AND JENSEN, D. Using of structure indices for efficinet approximation of network properties. In *Proc. of ACM SIGKDD* (Philadelphia, USA, 2006).

[26] RHEA, S., ET AL. Pond: The OceanStore prototype. In *Proc. of FAST* (April 2003).

[27] SWAMYNATHAN, G., ET AL. Do social networks improve e-commerce: a study on social marketplaces. In *Proc. of SIGCOMM WOSN* (August 2008).

[28] TANG, J., ET AL. Temporal distance metrics for social network analysis. In *Proc. of WOSN* (Barcelona, Spain, August 2009).

[29] TANG, L., AND CROVELLA, M. Virtual landmarks for the internet. In *Proc. of IMC* (Oct. 2003).

[30] VISWANATH, B., ET AL. On the evolution of user interaction in facebook. In *Proc. of SIGCOMM WOSN* (2009).

[31] WANG, C., KNIGHT, J. C., AND ELDER, M. C. On computer viral infection and the effect of immunization. In *Proc. of ACSAC* (2000).

[32] WILSON, C., BOE, B., SALA, A., PUTTASWAMY, K. P. N., AND ZHAO, B. Y. User interactions in social networks and their implications. In *Proc. of EuroSys* (April 2009).

[33] ZHENG, H., ET AL. Internet routing policies and round-trip times. In *Proc. of PAM* (April 2005).