

Thesis Summary:

Fast approximations of shortest paths and distance in large graphs and their applications

In large graphs, simple tasks such as finding the pairwise shortest path (PSP), or even just the distance between a pair of nodes in the graph, may consume too much resources. For example, if one wishes to set a continental route planning server (or Internet distance oracle) that should serve large volumes of queries - calculating the route (or delay) from a source to a destination junction (or router). For example, a route planning server (or an Internet distance estimation service), will be fairly slow because a continental road network contains over $20M$ junctions (or $300k$ Internet routers). In order to achieve near constant PSP query time, assuming the graph is updated periodically, e.g., every 10 minutes, we allow *batch preprocessing* of yielding pre-computed data used to accelerate the queries. Obviously, there is some tradeoff between query time, batch preprocessing time, and space for preprocessed information. In particular, for large graphs it would be prohibitive to preprocess and store shortest paths between all pairs of nodes. This work discusses two preprocessing approaches, geometric embedding and hierarchical, or multi-level, routing.

1 Embedding in Geometric Space

The network distance matrix can be compactly represented by mapping each network node to a point in a geometric space. Such a mapping, called *embedding*, is designed to preserve the network distance between any pair of network nodes, so that the estimated pair distance is given by the geometric distance between the representing two points. The *storage complexity* of the embedding coordinates in a d -dimensional space is dn , and the query time is $O(d)$, where d is the dimension of the geometric space. Unlike shortest path based methods, embedding estimates the distance to any destination by measuring just a small number of distances, which is advantageous, for example, for application level nodes that are unaware of the underlying physical topology of the Internet.

Big-Bang Springs (BBS) We present a new scheme for embedding, based on a novel idea, utilizing notions from Newtonian mechanics. BBS models the network

nodes as a set of particles, each is the 'geometric image' of a node, that is the position of this particle in Euclidean space. The particles are traveling in that space under the effect of potential force field. The force field reduces the potential energy of the particles, that is related to the total embedding error of all particle pairs. Each pair of particles is pulled or repulsed by the field force induced between them depending on their pair embedding error, that is the embedding error of the distance between them. As a particle accelerates under the effect of the force field it is also attenuated by simulated friction force.

We compare our scheme to other embedding methods and show that it produces the best embedding over various parameter choices with reasonable complexity. This is not to say that for special cases, e.g., when the number of embedded nodes is very small, it will outperform all other schemes, but it will be better than any other scheme when all cases are considered. The BBS scheme advantage over conventional gradient minimization schemes, such as steepest decent and down-hill simplex (DHS), is that the kinetic energy accumulated by the moving particles enables them to escape local minima. Moreover, DHS which was used by Ng and Zhang's GNP is very sensitive to the initial vertices coordinates.

Internet Embedding in Hyperbolic Space In the Internet due to BGP policy routing, many of routes tend to path through the network core, and not to take available tangential links. Thus, we suggest an algorithm which embeds the Internet graph into a hyperbolic space with preselected *curvature*, where the line between two points in the Internet periphery bends towards the origin of the hyperbolic space, the area in which the core of the Internet is embedded, and thus follows the true Internet route.

We embed the metric in a d -dimensional hyperbolic target space of varying curvature values, and deduct the optimal curvature by comparing their distance error results. We found that the sensitivity to the exact curvature value is mild, and improve the performance of three applications: delay estimation (which can be used for QoS threshold estimation), server selection, and application level multicast.

2 Multi-level Proximity Routing (MPR)

We introduce MPR, a multi-level heuristic algorithm that finds the PSP by discovering the hierarchal routing structure from the unlabeled input, containing just the graph nodes and the weights of their connecting arcs. We use the Algebraic Multi Grid paradigm, customized to the shortest-path routing problem, to calculate the MPR structure, a hierarchical soft clustering structure in which each cluster can belong to one *or more* next-level clusters. The MPR nodes are the nodes of the given input graph. Each hierarchy node, except for the root, has one or *several* parents, and the set of arcs from children to their parents form a directed acyclic graph (DAG), instead of a tree for hard clustering. Each DAG arc (s, t) corresponds to an input path, and the

weight of the arc (s, t) is equal to the path weight. The MPRQuery algorithm, given any source and destination nodes, efficiently finds the shortest MPR path between the two nodes and returns its cost as their approximated pairwise shortest path distance. Initially, each network node forms a $l = 1$ level cluster, and the network edges are $l = 1$ level edges between these clusters. Next, the *batch preprocessing* step, calculates the $(l + 1)$ level from the l 'th level.

Based on the above MPR algorithm we develop ϵ -MPR, a hierarchical provably bounded ϵ error algorithm for the PSP problem. Like MPR, the ϵ -MPR is a multi-level algorithm which applies its preprocessing step repeatedly. In each ϵ -approximated preprocessing step we inspect all the 2- and l -level hop paths via 2 neighboring neglected nodes. For each such path we find an alternative path in the constructed MPR hierarchy approximating the weight of the neglected path up to a factor $(1 + \epsilon)$. We prove that the paths found by the above MPRQuery algorithm in the resulting ϵ -MPR structure have a stretch $T(\epsilon)$ for small ϵ .

We tested the versatility of the two algorithms with three kinds of large physical networks, the Internet IP graph measured by the DIMES project, a generated Bay-area Ad Hoc network and the Europe road network. The Internet graph has a power-law degree distribution compared to the near constant degree distribution of road networks, and the generated Ad Hoc graph is 5.5 times denser than the Europe road graph (average node degree 12.61 compared to 2.3). Nevertheless, our basic MPR performed equally well for all three graphs, with exactly the same parameters. For sparse graphs such as the Europe road network, our basic MPR is faster (2.37 and 2.75 times faster preprocessing and query time respectively) than HHs, the fastest known multi-level preprocessing algorithm. The maximum relative error in the experiments of ϵ -MPR with the above datasets, is 1.24.