CSE 6441: Project Report on Betweenness Centrality

Definition:Betweenness Centrality marks the importance of nodes by their participation in all shortest paths of network.

For a graph G of Edges m and Vertices n

Algorithms and Time Complexity:

1) Static Exact Algorithms:

Brandes Algorithm is the fastest existing algorithm for exact computation of Betweenness Centrality,it takes O(mn) for unweighted graphs and O(nm+n^2logn) for graphs with positive edge weights.

The algorithm computes for every node s ∈ V a slightly-modified version of a single-source shortest-path tree (SSSP tree), producing for each s the directed acyclic graph (DAG) of all shortest paths starting at s. Exploiting the information contained in the DAGs, the algorithm computes the dependency δs(v) for each node v, that is the sum over all nodes t of the fraction of shortest paths between s and t that v is internal to. The betweenness of each node v is simply the sum over all sources s ∈ V of the dependencies δs(v). Therefore, we can see the dependency δs(v) as a contribution that s gives to the computation of cB(v).

Based on this concept many approximation algorithms have been proposed

2)Static Approximation Algorithms:

Brandes and Pich propose to approximate cB(v) by extrapolating it from the contributions of a subset of source nodes, also called pivots.
Selecting the pivots uniformly at random, the approximation can be proven to be an unbiased estimator for cB(v) (i.e. its expectation is equal to cB(v)).

In a subsequent work, Geisberger notice that this can produce an overestimation of BC scores of nodes that happen to be close to the sampled pivots. To limit this bias, they introduce a scaling function which gives less importance to contributions from pivots that are close to the node.
Bader et al, approximate the BC of a specific node only, based on an adaptive sampling technique that reduces the number of pivots for nodes with high centrality.
Different from the previous approaches is the approximation algorithm of Riondato and Kornaropoulos, which samples a single random shortest path at each iteration. This approach allows theoretical guarantees on the quality of their approximation: For any two constants , $\delta \in (0, 1)$, a number of samples can be defined such that the error on the approximated values is at most with probability at least $1 - \delta$.

3)Exact Dynamic Algorithms: Dynamic algorithms update the betweenness values of all nodes in response to a modification on the graph, which might be an edge insertion, an edge deletion or a change in an edge's weight.

QUBE by Lee relies on the decomposition of the graph into connected components. When an edge update occurs, QUBE re-computes the centrality values using Brandes Algorithm only within the affected component. In case the update modifies the decomposition, this must be recomputed, and new centralities must be calculated for all affected components.

The approach proposed by Green for unweighted graphs maintains a structure with the previously calculated BC values and additional information, like the distance of each node v from every source $s \in V$ and the list of predecessors, i.e. the nodes immediately preceding v in all shortest paths from s to v. Using this information, the algorithm tries to limit the re-computations to the nodes whose betweenness has actually been affected.

Kourtellis modified the approach proposed by Green in order to reduce the memory requirements. Instead of storing the predecessors of each node v from each possible source, they recompute them every time the information is required.

Kas extended an existing algorithm for the dynamic APSP problem by Ramalingam and Reps to also update BC scores.

The recent work by Nasre contains the first dynamic algorithm for BC (NPR) which is asymptotically faster than recomputing from scratch on certain inputs. In particular, when only edge insertions are allowed and the considered graph is sparse and weighted, their algorithm takes O(n^2) operations, whereas BA requires O(n^2 log n) on sparse weighted graph.

These are some of the existing algorithms to solve the problem.

Scope of Parallelization: As the calculation of Betweenness centrality depends mainly on calculation of single source shortest path.SSSP can be parallelized by different approaches discussed in class, like source partition or source parallel.

In the Brandes algorithm,SSSP should be calculated for each node, so SSSP for each node can be processed parallely on a different processor followed by subsequent reduction to calculated Betweenness centrality.