MADALGO CENTER FOR MASSIVE DATA ALGORITHMICS

Andreas Beckmann
Goethe University

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

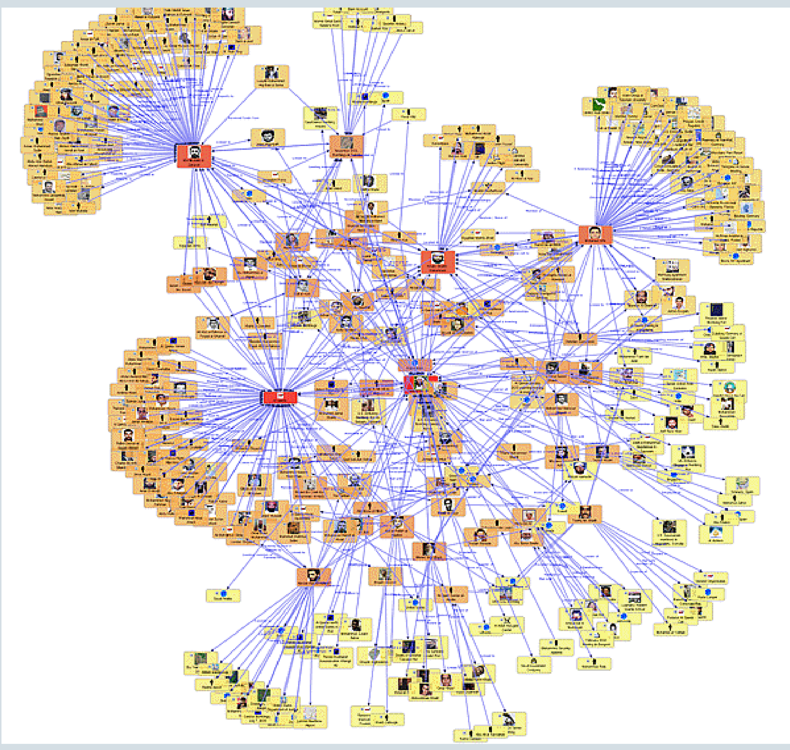
Deepak Ajwani
Aarhus University

Approximating the Diameter of Large Graphs

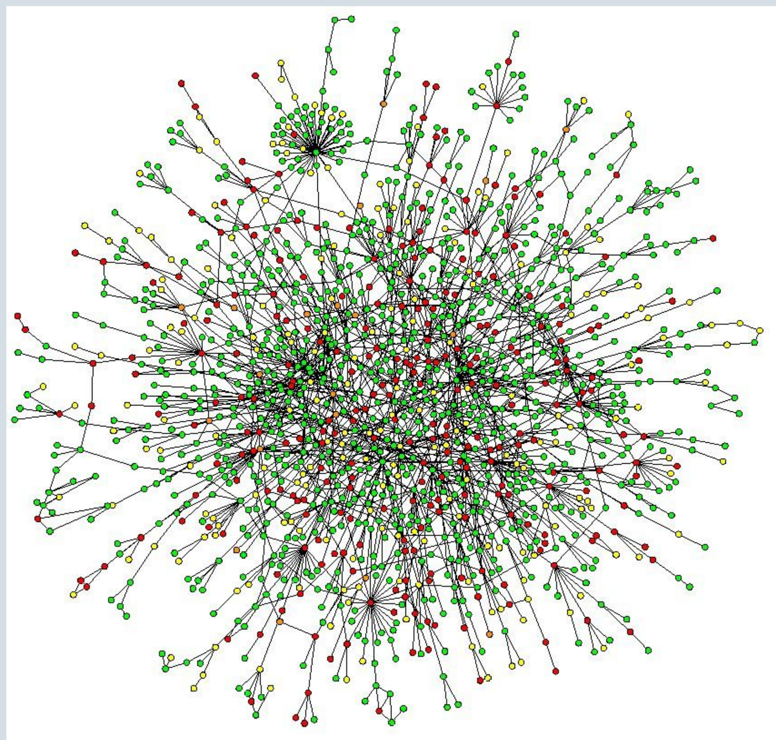
Large Graphs

Large graphs arise naturally in many application domains, such as network analysis

Social Networks

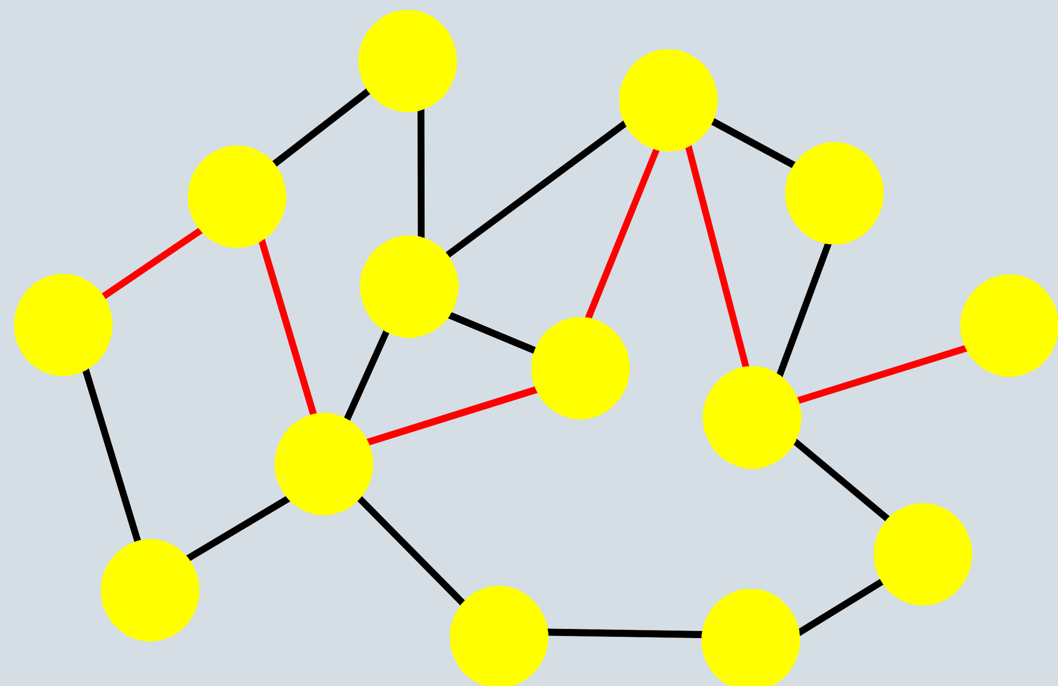


Yeast Protein Interaction Network



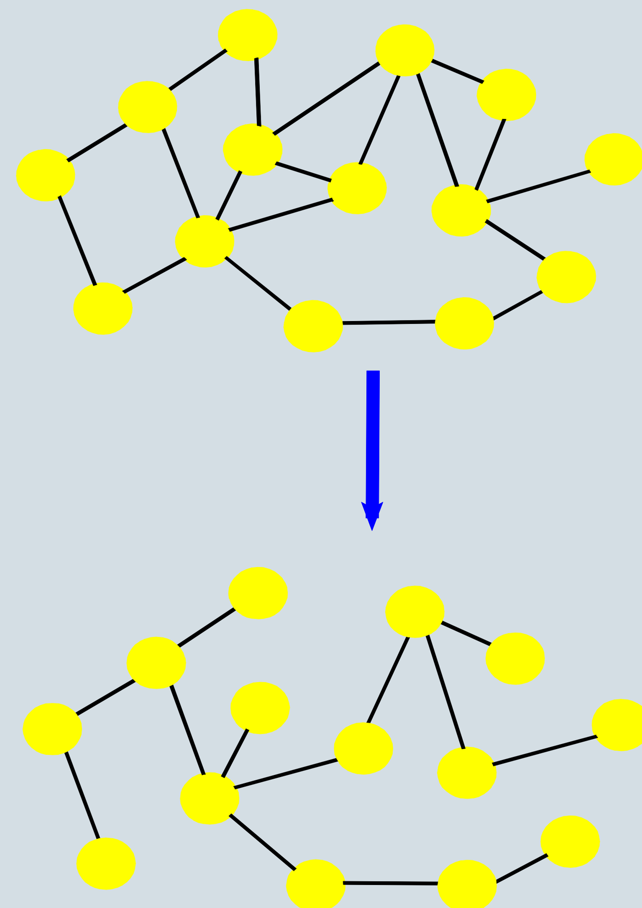
Diameter of a Graph

The diameter is the length of the longest shortest path between any two nodes in a graph. For instance, in the graph below, diameter = 6



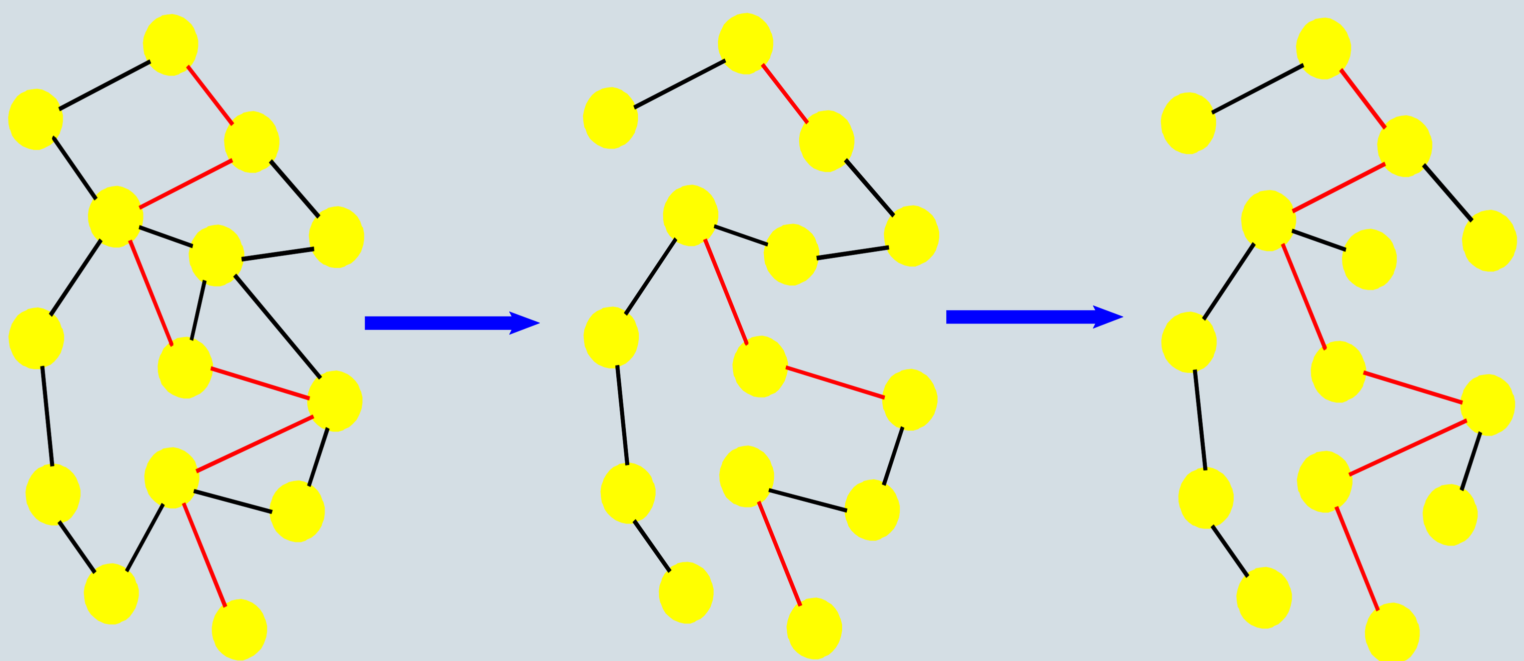
Computing the diameter is a fundamental step in network analysis

Breadth-First Search Tree



- Height of a Breadth-First Search tree gives a 2-approximation to the diameter of the graph
- Computing Breadth-First Search requires days for large graphs with billions of edges, even with the most efficient external memory algorithms
- Need to approximate the diameter even faster than the Breadth-First Search traversal of the graph
- We perform an empirical study to compare the running time and approximation quality of different algorithms

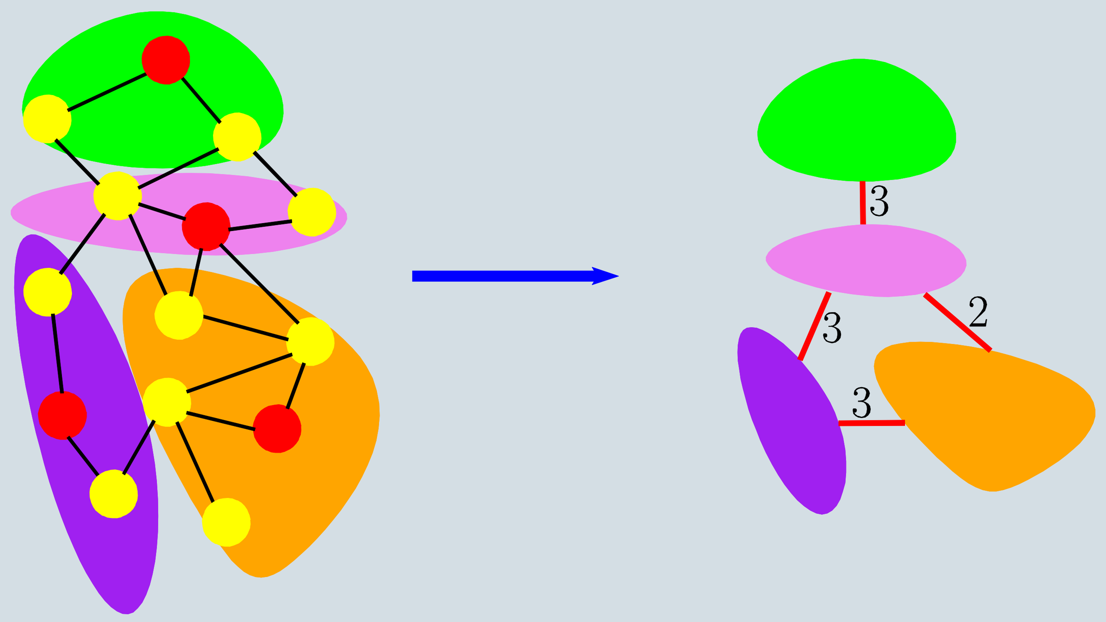
Heuristics to Reduce Spanning Tree Diameter



Input Graph — Minimum Spanning Tree — Spanning Tree after 1st Iteration

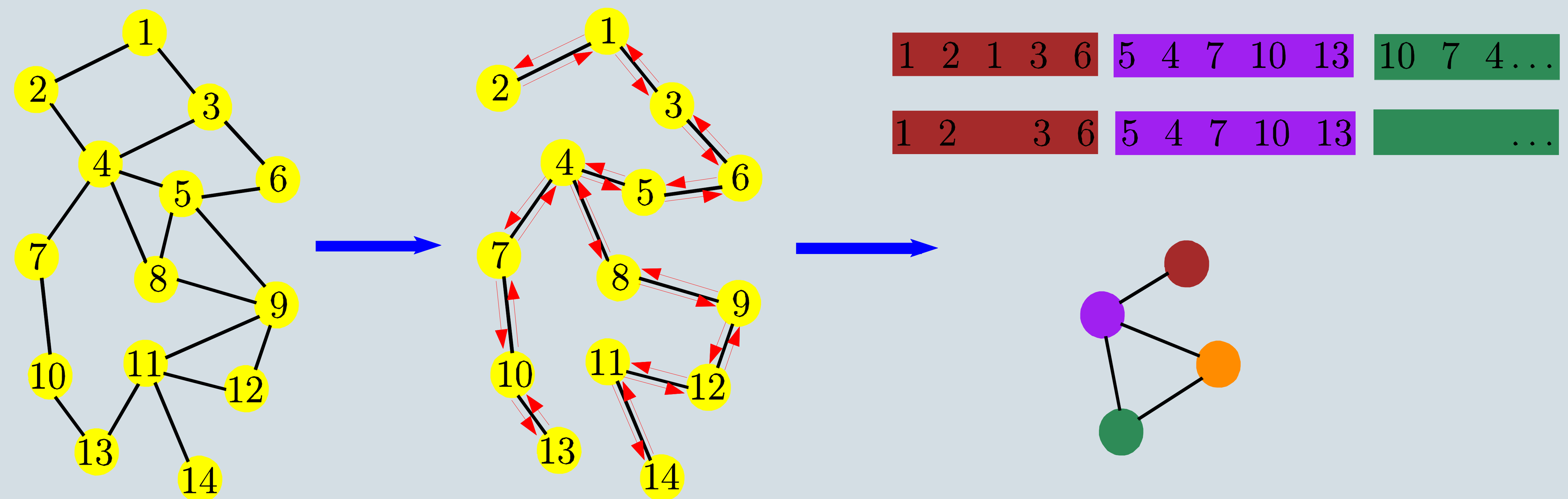
- Assign random weights to the graph edges
- Compute a minimum spanning tree
- For each node, select an incident edge to the lowest depth node
- Selected edges form a new spanning tree
- Repeat the above iteration till some convergence threshold

Parallel Cluster Growing Algorithm



- Choose n/k random master nodes plus O(n/k) deterministic ones (Euler tour)
- Grow clusters around master nodes in parallel (local BFS runs)
- Each node gets labelled with its cluster index and distance from its master after O(k) phases
- Form a new graph G’
  - Each node represents a cluster in the original graph
  - Edge {C(u), C(v)} exists in G’ if the edge {u, v} exists in G
  - Edge weight: dist(u,u’) + 1 + dist(v,v’), where u’ and v’ are master nodes in C(u) and C(v), respectively
- Compute single-source shortest paths from an arbitrary node in the smaller graph G’

Euler Tour Based Algorithm



- Compute a spanning tree of the input graph
- Build an Euler tour around it and chop it into O(n/k) clusters of length k
- Eliminate duplicates by keeping only the first occurrence
- Form a new (smaller) graph G’
  - Each node represents a non-empty cluster in the original graph
  - Edge {C(u), C(v)} exists in G’ if the edge {u, v} exists in G
- Run Breadth-First Search on G’ starting from an arbitrary node

Preliminary Results

- The heuristic approach is quite fast and requires only 2-3 iterations to get a 1.5-approximation of graph diameter
- With multiple disks, the heuristic approach becomes computation bound

References

[1] Ulrich Meyer. *On Trade-Offs in External-Memory Diameter-Approximation*. SWAT, 2008.

MADALGO – Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation