MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS

Deepak Ajwani
Bell Labs, Ireland

Ulrich Meyer
Goethe University

David Veith
Goethe University

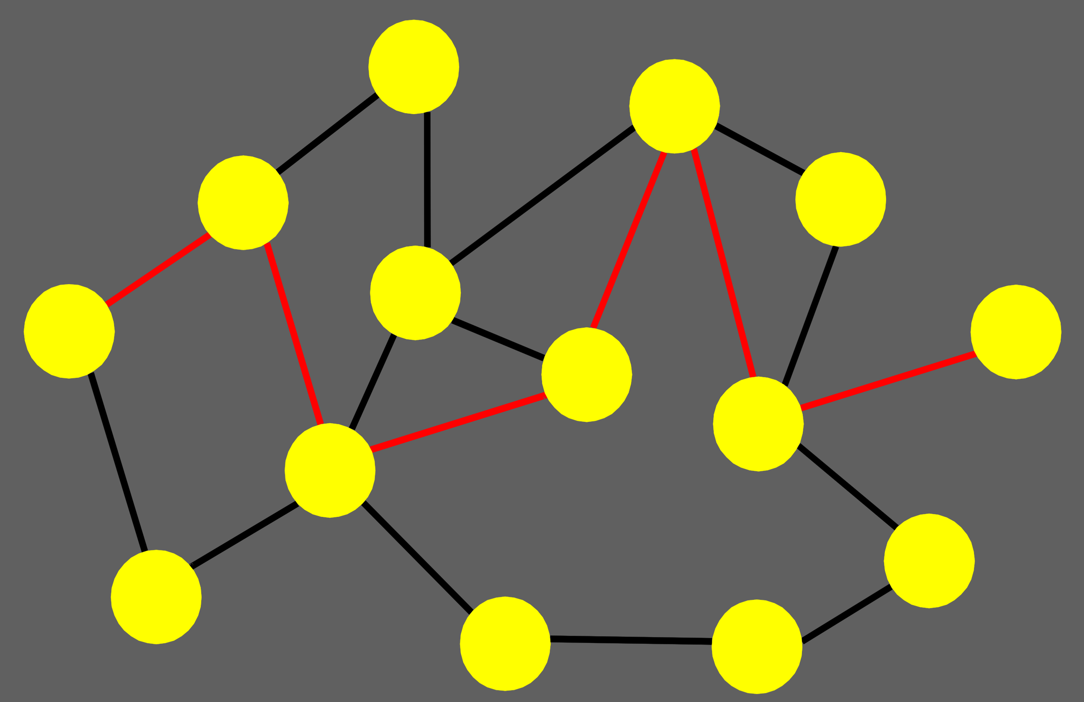
GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Danmarks Grundforskningsfond
Danish National Research Foundation

I/O-efficient Hierarchical Diameter Approximation

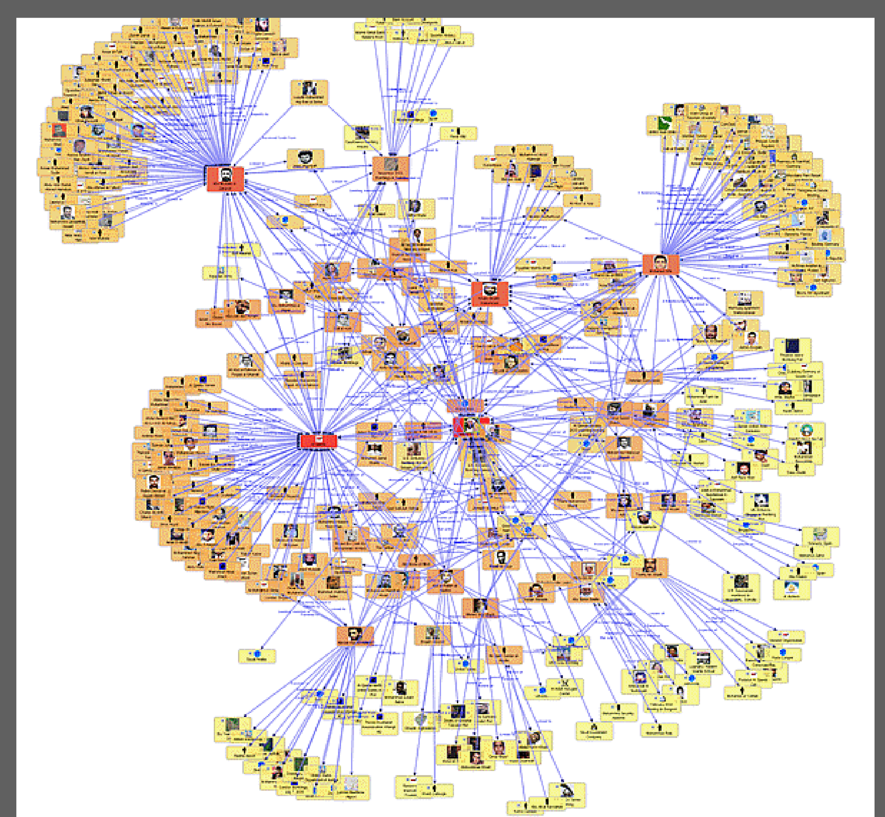
Diameter Approximation

We want to approximate the diameter of large graphs. The diameter is the length of the longest shortest path between any two vertices in a graph.



The diameter of this graph is determined by the red path

One application of the diameter approximation is to select suitable algorithms for BFS computation. Knowing the diameter the number of I/Os can be reduced in some cases (in particular for real world graphs like web or social network graphs which usually have a small diameter).



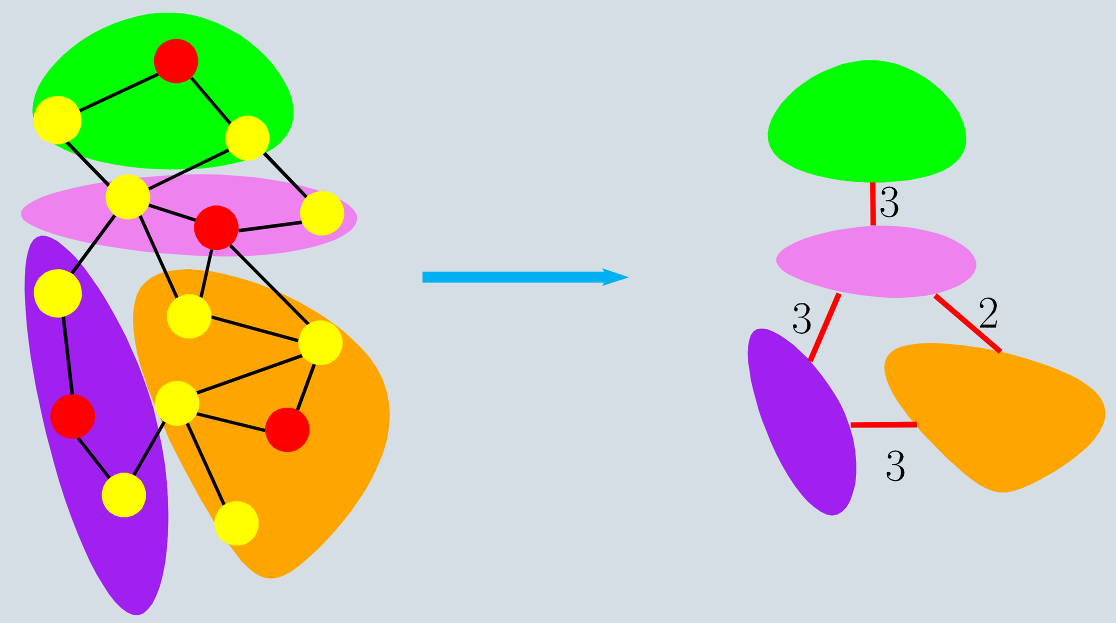
Social networks usually have a small diameter in O(log(N))

Some important facts

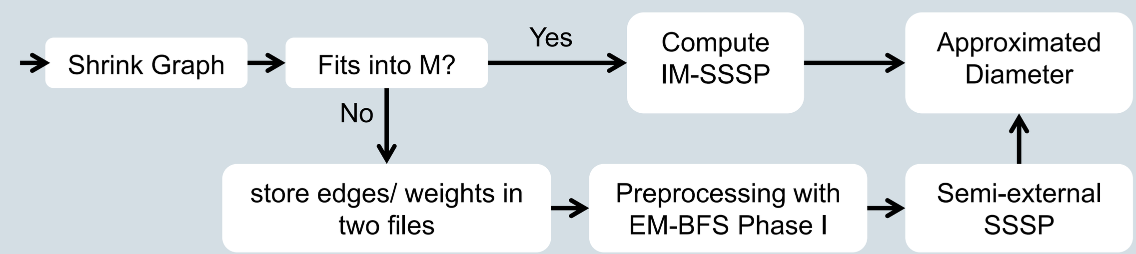
- Computation of the exact diameter in main memory is very expensive with $O(N^2)$.
- Approximation using a single BFS with approximation factor ½.
- The I/O complexity of external-memory BFS is $\Omega(N/\sqrt{B})$ I/Os.
- Trade-off: Diameter approximation using Parallel Cluster Growing Algorithm [1] with $O(k \cdot scan(N) + sort(N) + N \cdot \sqrt{\log(k)/(k \cdot B)})$ I/Os.

Parallel Cluster Growing Algorithm

To approximate the diameter with less I/Os than BFS we shrink the input size by selecting randomly O(N/k) master vertices with probability O(1/k) and run local BFS in parallel. Each master vertex is a source of such a local BFS.

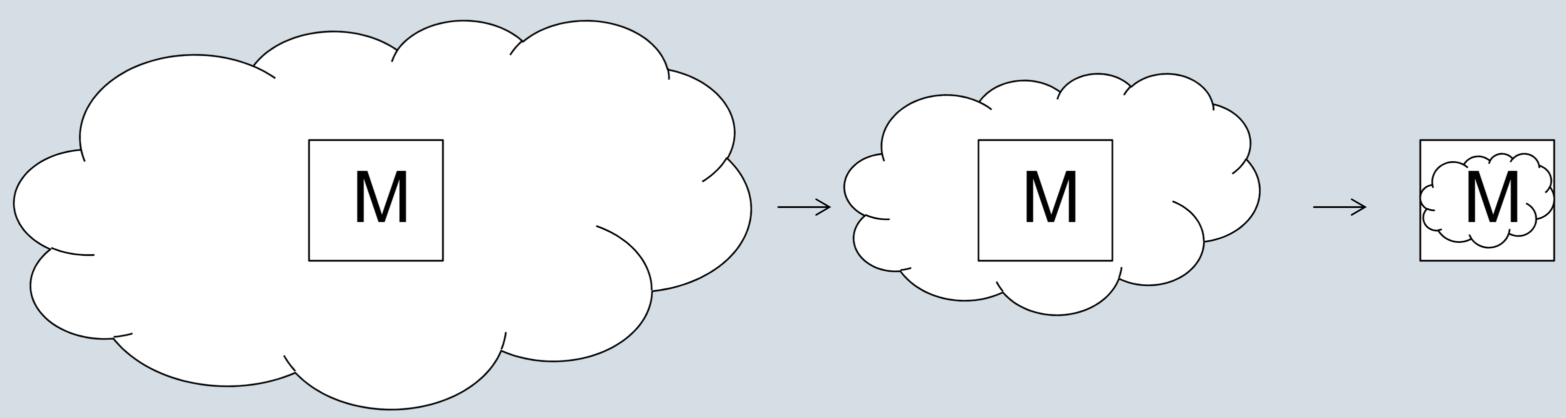


After clustering we run a single-source shortest path. We try to shrink the graph so that it fits into main memory. However, a small amount of master vertices can result into a large I/O complexity for graph classes with larger diameter [2]. To avoid this negative behaviour we decided to develop a recursive version.



Hierarchical Diameter Approximation

- Uses main idea from parallel cluster growing but shrinks the graph several times (in our experiment two shrinking steps were enough even for 128 GB graph data).
- The worst case guarantee is now $\Omega(k^{4/3-\varepsilon})$ instead of $O(k^{1/2} \cdot \log(k))$.
- Experimental results are much better than the pessimistic bound predicts [3].
- To improve our results we developed some heuristics.

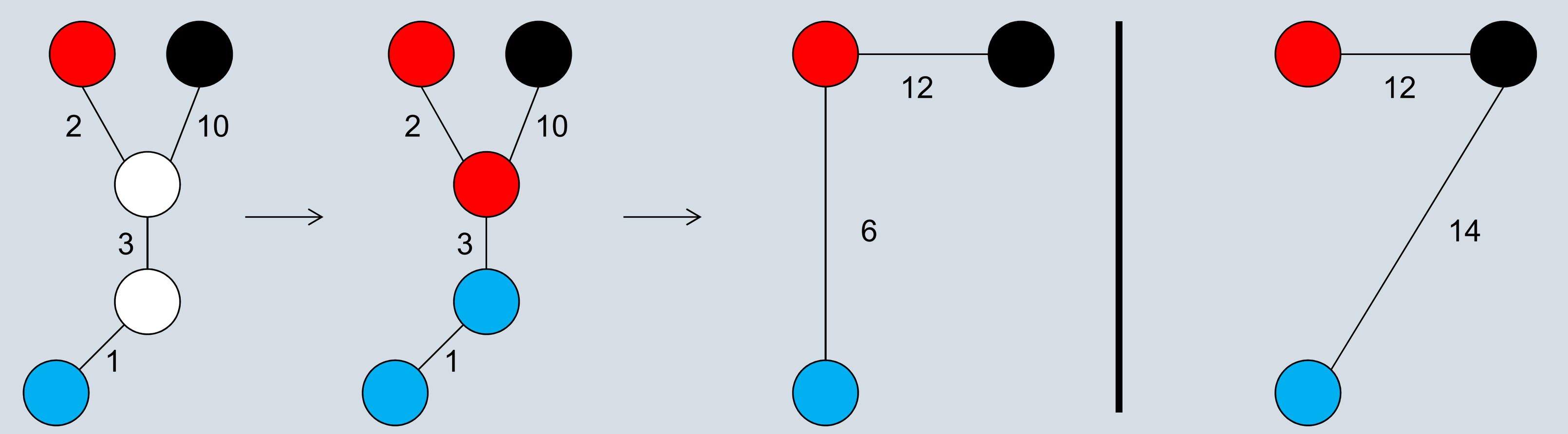


Main memory M is too small for the big data. Therefore shrink the data until it fits into the main memory for fast computation.

Further Improvements & Results

For the hierarchical diameter approximation we have developed several heuristics to improve the result:

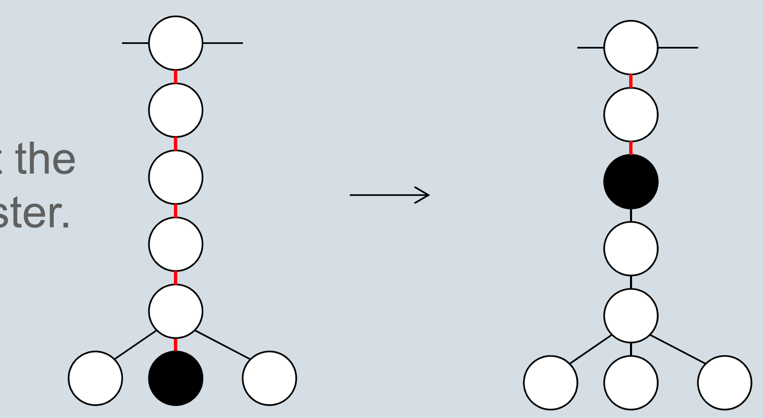
- Break ties that the path length is kept small in the shrunken graph



Without TIE breaking

- Move masters to the cluster centre to counterbalance weights

The master is at the border of its cluster.



The master is in the centre of its cluster. So, we achieve counterbalanced distances to all adjacent clusters.

Results

	Size [GB]	HIER-time	ratio	DSLB-BFS-time (single)	ratio
web graph	27	0.6 h	3.3	5.6 h (3.2 h)	~ 1.0
√n-level graph	128	9.0 h	~1.0	56.6 h (37.3 h)	1.0
n-level graph	83.8	3.6 h	~1.0	27.8 h (19.0 h)	1.0
worse 2step	31.9	1.4 h	3.1	15.3 h (11.6 h)	1.0

References

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

[2] Deepak Ajwani, Andreas Beckmann, Ulrich Meyer, David Veith. *I/O-efficient approximation of graph diameters by parallel cluster growing – a first experimental study*. PASA 2012.

[3] Deepak Ajwani, Ulrich Meyer, David Veith. *I/O-efficient Hierarchical Diameter Approximation*. ESA 2012.