Determining the Diameter of Small World Networks

Frank W. Takes & Walter A. Kosters

Leiden University, The Netherlands

CIKM 2011 — October 27, 2011 — Glasgow, UK

NWO COMPASS project (grant #612.065.92)

Overview

Universiteit Leiden

- Introduction
- Preliminaries
- Problem statement
- Related work
- Algorithm
- Results
- Conclusion
- Future work

Introduction

Universiteit Leiden

- Small world networks
- Power law degree distribution, giant component, low average pairwise distances
- Examples: social networks, webgraphs, communication networks, collaboration networks, information networks, protein-protein interaction networks, citation networks, etc.
- **Diameter**: length of longest shortest path in a graph

Diameter Example

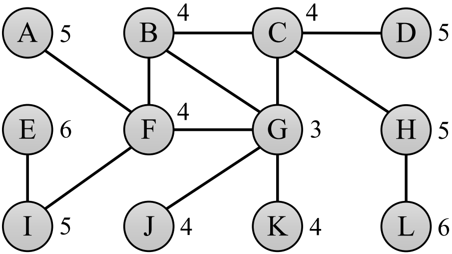


Figure: Graph with diameter 6. Numbers denote node eccentricity

Preliminaries

Universiteit Leiden

- Graph $G = (V, E)$ with $|V| = n$ nodes and $|E| = m$ edges
- Distance $d(u, v)$: length of shortest path between $u, v \in V$
- Undirected: $d(u, v) = d(v, u)$ for all $u, v \in V$
- One connected component: $d(u, v)$ is finite for all $u, v \in V$
- Neighborhood $N(u)$: set of nodes connected to u via an edge
- Degree $deg(u)$: number of edges connected to node u

Problem statement

Universiteit Leiden

- Consider a connected undirected graph $G = (V, E)$
- Our aim is to compute in large small-world graphs $(1,000 \leq n \leq 100,000,000, \overline{d}(u, v) \approx 6$ for all $u, v \in V)$:

Problem statement

Universiteit Leiden

- Consider a connected undirected graph $G = (V, E)$
- Our aim is to compute in large small-world graphs ($1,000 \leq n \leq 100,000,000$, $\overline{d}(u, v) \approx 6$ for all $u, v \in V$):
  - **Eccentricity** $e(v)$: length of a longest shortest path from v: $e(v) = \max_{w \in V} d(v, w)$
  - **Diameter** $D(G)$: maximal distance (longest shortest path length) over all node pairs: $\max_{v,w \in V} d(v, w)$

Problem statement

- Consider a connected undirected graph $G = (V, E)$
- Our aim is to compute in large small-world graphs $(1,000 \leq n \leq 100,000,000,\ \overline{d}(u, v) \approx 6$ for all $u, v \in V)$:
  - **Eccentricity** $e(v)$: length of a longest shortest path from v: $e(v) = \max_{w \in V} d(v, w)$
  - **Diameter** $D(G)$: maximal distance (longest shortest path length) over all node pairs: $\max_{v,w \in V} d(v, w)$
  - **Diameter** $D(G)$ (alternative definition): maximal eccentricity over all nodes: $\max_{v \in V} e(v)$

Problem statement

Universiteit Leiden

- Consider a connected undirected graph $G = (V, E)$
- Our aim is to compute in large small-world graphs $(1,000 \leq n \leq 100,000,000,\ \overline{d}(u,v) \approx 6$ for all $u, v \in V)$:
  - **Eccentricity** $e(v)$: length of a longest shortest path from v: $e(v) = \max_{w \in V} d(v, w)$
  - **Diameter** $D(G)$: maximal distance (longest shortest path length) over all node pairs: $\max_{v,w \in V} d(v, w)$
  - **Diameter** $D(G)$ (alternative definition): maximal eccentricity over all nodes: $\max_{v \in V} e(v)$
  - **Radius** $R(G)$: minimal eccentricity over all nodes: $\min_{v \in V} e(v)$
  - Eccentricity distribution: (relative) frequency $f(x)$ of each eccentricity value x

$$f(x) = \frac{|\{u \in V \mid e(u) = x\}|}{n}$$

Eccentricity distribution

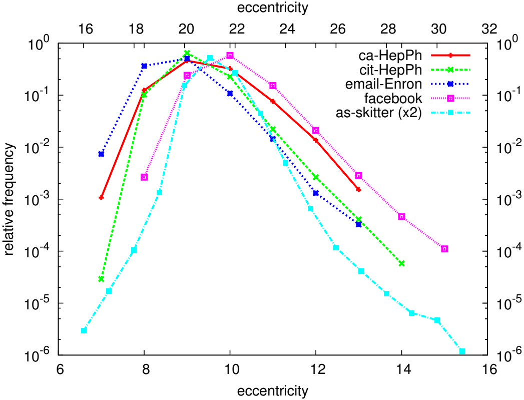


Figure: Relative eccentricity distribution of five large graphs

Diameter Applications

- Router networks: what is the **worst-case** response time between any two machines?
- Social networks: in how many steps does a message released by a single user reach **everyone** in the network?
- Biological interaction networks: which proteins are likely to not influence each other at all?
- Information networks (i.e., Wikipedia): how do I change the conversation topic to a **maximally different** subject? ;-)
- Eccentricity has been suggested as a worst-case measure of **node centrality**: the relative importance of a node based on the graph's structure

Naive Algorithm

Universiteit Leiden

- Diameter is equal to the largest value returned by an **All Pairs Shortest Path** (APSP) algorithm
- Brute-force: for each of the n nodes, execute a **Breadth First Search** (BFS) run in $O(m)$ time to find the eccentricity, and return the largest value found
- Time complexity $O(mn)$
- Problematic if $n = 8$ million and $m = 1$ billion.
  Then one BFS takes 6 seconds on a 3.4GHz machine.
  That results in 1.5 years to compute the diameter ...

Related work

Universiteit Leiden

- Approximation algorithms, for example ANF (Palmer et al.)
- Use a random sample of the set of nodes (Mislove et al.)
- Heuristics, for example repeatedly select the farthest node until there is no more improvement (Leskovec et al.)
- Matrix multiplication for APSP in $O(n^{2.376})$ (Yuster et al.)
- Bounds: diameter upper bound is at most two times the lowest found eccentricity value (Magnien et al.)

Social Network Example (1)

Universiteit Leiden

- If I am connected to everyone in at most 6 steps, then
  - My direct friend is connected to everyone in at most 7 steps (he reaches everyone through me)
  - My direct friend is connected to everyone in at least 5 steps (I reach everyone through him)

- If I can reach everyone in the network in 6 steps, then
  - There is nobody who can reach everyone in less than 3 steps (or I could have utilized him)
  - There is nobody who needs more than 12 steps to reach everyone
    (or he could have utilized me)

Social Network Example (2)

- If a node v has eccentricity $e(v)$, then
  - Nodes w at distance $d(v, w)$ needs at most $e(v) + d(v, w)$ steps
    (w reaches every node via v)
  - Nodes w at distance $d(v, w)$ needs at least $e(v) - d(v, w)$ steps (v reaches every node via w)

  We call this the **Eccentricity bounds**
- If a node v can reach every other node in $e(v)$ steps, then
  - There is no node that can reach everyone in less than $\lceil e(v)/2 \rceil$ steps (or v could have used that node)
  - There is no node that needs more than $e(v) \cdot 2$ steps to reach all other nodes (or that node could have used v)

  We call this the **Diameter bounds**

Eccentricity bounds

Universiteit Leiden

Theorem

For nodes $v, w \in V$ we have
$\max(e(v) - d(v,w), d(v,w)) \leq e(w) \leq e(v) + d(v,w)$.

Proof

- Upper bound $e(v) + d(v,w)$: if node w is at distance $d(v,w)$ of node v, it can always employ v to get to every other node in $e(v)$ steps. To get to node v, exactly $d(v,w)$ steps are needed, totalling $e(v) + d(v,w)$ steps to get to any other node.
- Lower bound $e(v) - d(v,w)$: interchanging v and w in the previous statement.
- Lower bound $d(v,w)$: the eccentricity of w is at least equal to some found distance to w.

Diameter bounds

- Let $e_L(v)$ and $e_U(v)$ denote the lower and upper eccentricity bounds derived using the *Eccentricity bounds*.
- Then we can derive the following **diameter bounds**: $\max_{v \in V} e_L(v) \leq D(G) \leq \max_{v \in V} e_U(v)$
- Let $D_L(G)$ and $D_U(G)$ denote these lower and upper diameter bounds. $D_L(G) \leq D(G) \leq D_U(G)$

BoundingDiameters Algorithm

Universiteit Leiden

Input: Graph G
Output: Diameter of G

$W \leftarrow V \qquad D_\ell \leftarrow -\infty \qquad D_u \leftarrow +\infty$
for $w \in W$ **do**
    $e_\ell[w] \leftarrow -\infty$
    $e_u[w] \leftarrow +\infty$
end for

while $D_\ell \neq D_u$ **and** $W \neq \emptyset$ **do**
    $v \leftarrow \textsc{SelectFrom}(W)$
    $e[v] \leftarrow \textsc{Eccentricity}(v)$

    $D_\ell \leftarrow \max(D_\ell, e[v])$
    $D_u \leftarrow \min(D_u, 2 \cdot e[v])$

    for $w \in W$ **do**
        $e_\ell[w] = \max(e_\ell[w], \max(e[v] - d(v, w), d(v, w)))$
        $e_u[w] = \min(e_u[w], e[v] + d(v, w))$
        if $(e_u[w] \leq D_\ell$ **and** $e_\ell[w] \geq D_u/2)$ **or**
            $(e_\ell[w] = e_u[w])$ **then**
                $W \leftarrow W - \{w\}$
        end if
    end for

    $D_u \leftarrow \min\left(D_u, \max_{w \in V}(e_u[w])\right)$
end while

return D_ℓ;

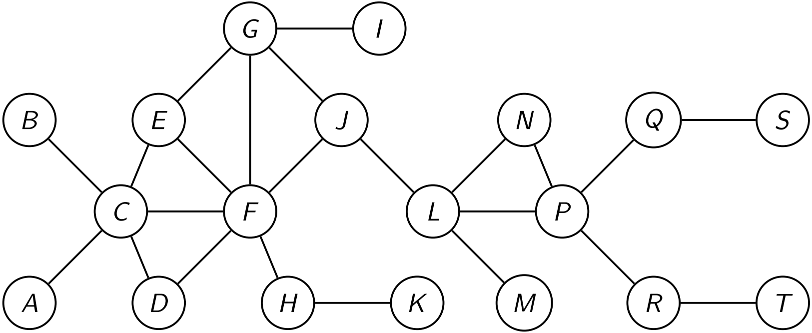
BoundingDiameters Algorithm

Universiteit Leiden

- Initialize candidate set W to V
  While $D_L(G) \neq D_U(G)$:
  1. Select a node v from W cf. some **Selection strategy**
  2. Compute v's eccentricity, and update $e_L(v)$ and $e_U(v)$ for every node $v \in W$ according to the **Eccentricity bounds**
  3. Update the diameter bounds $D_L(G)$ and $D_U(G)$
  4. Remove nodes w that can no longer contribute to refining the **Diameter bounds**
- Worst-case: n iterations, best-case: 2 iterations (investigate v and w with $e(v) = 2 \cdot e(w) = D(G)$)
- To compute the complete eccentricity distribution, stop when: $\forall v \in V : e_L(v) = e_U(v)$
- **Selection strategy** is important (and discussed later)

Example run (0)

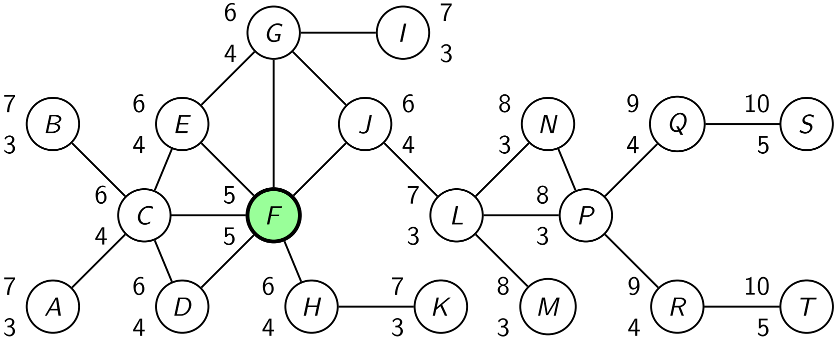
Universiteit Leiden



What is the diameter of this graph?

$D_L = -\infty$ and $D_U = \infty$

Example run (1)

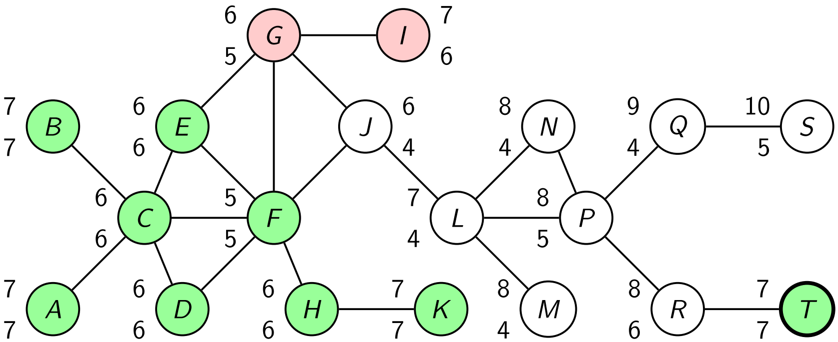


Iteration 1: after computing the eccentricity of node F

$D_L = 5$ and $D_U = 10$

Example run (2)

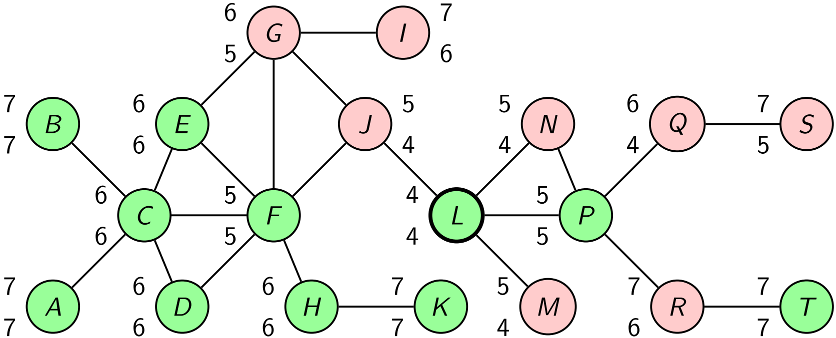
Universiteit Leiden



Iteration 2: after computing the eccentricity of node T

$D_L = 7$ and $D_U = 10$

Example run (3)



Iteration 3: after computing the eccentricity of node L

$D_L = 7$ and $D_U = 7$

Selection strategy

Universiteit Leiden

- Random node ("smart APSP")
- Based on the degree of the node
- Eccentricity bounds difference (1)
- **Interchange smallest eccentricity lower bound and largest eccentricity upper bound (2)**
- Repeated farthest distance (cf. Leskovec et al.) (3)

Results

1. Eccentricity bounds difference
2. **Alternate between smallest eccentricity lower bound and largest upper bound**
3. Repeatedly select a node furthest away from the previous node

Dataset	Nodes	$D(G)$	Strat. 1	Strat. 2	Strat. 3	Pruned
AstroPhys	17,903	14	18	**9**	63	185
Enron	33,696	**13**	12	**11**	61	8,715
Web	855,802	24	20	**4**	28	91,965
YouTube	1,134,890	24	2	**2**	2	399,553
Flickr	1,624,992	24	10	**3**	7	553,242
Skitter	1,696,415	**31**	10	**4**	19	114,803
Wikipedia	2,213,236	18	21	**3**	583	947,582
Orkut	3,072,441	10	357	**106**	389	27,429
LiveJournal	5,189,809	23	6	**3**	14	318,378
Hyves	8,057,981	**25**	40	**21**	44	446,258

Table: Comparison of three node selection strategies

Pruning

Theorem

Assume $n > 2$. For a given $v \in V$, all nodes $w \in N(v)$ with $deg(w) = 1$ have $e(w) = e(v) + 1$.

Proof

- Node w is only connected to node v, and will thus need node v to reach every other node in the graph. If node v can do this in $e(v)$ steps, then node w can do this is in exactly $e(v) + 1$ steps.
- The restriction $n > 2$ on the graph size excludes the case in which w realizes the eccentricity of v.

(alternative proof is possible, based on graph homomorphism)

Discussion

Universiteit Leiden

- **Main result:** in real-world graphs BoundingDiameters is much faster than the naive algorithm (a handful vs. n BFSes)
- **Why does it work?** There is always diversity in the eccentricity values of nodes, allowing central nodes to influence the eccentricity of peripheral nodes, and vice versa
- **When does it not work so well?** In graphs with little diversity in the eccentricity values, e.g., circle-shaped graphs
- **Side result**: efficiently computing derived measures such as the radius, center, periphery and even the exact eccentricity distribution is also possible (after some modifications)

Conclusion

Universiteit Leiden

- Our algorithm computes the diameter of large real world graphs much faster compared to the naive algorithm.
- Our algorithm improves upon previously suggested techniques, because:
  - we obtain an exact result instead of an approximation
  - it is possible to obtain the actual diameter path
  - information between iterations is not thrown away
  - computation time is very short, even for graphs with millions of nodes
- Future work: optimize the node selection strategy even further and incremental udates as the graph changes over time through the addition and deletion of nodes and edges.

Questions?

Universiteit Leiden