On Computing the Diameter of Real World Graphs

Andrea Marino

PhD Course on Graph Mining Algorithms,
Università di Pisa

Pisa, February 2018

Small world effect

The small average distance observed in the complex networks is referred as small world effect:

- If the size of the network is n, the average distance and the diameter have at most the order of magnitude of $\log(n)$.

The average distance in Facebook (721.1M nodes and 68.7G edges) is 5.7 and the diameter is 41.


The New York Times

Business Day

Technology

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH

M·COIN Mobile payments

75 COUNT

Separating You and Me? 4.74 Degrees

By JOHN MARKOFF and SOMINI SENGUPTA

Published: November 21, 2011

- Average distance has been computed by applying HyperANF tool.
  - Boldi, Rosa, and Vigna. Hyperanf: approximating the neighbourhood function of very large graphs on a budget. In WWW 2011.
- Diameter has been computed by applying *i*FUB.
  - Crescenzi, Grossi, Habib, Lanzi, and Marino. On computing the diameter of real-world undirected graphs. Theoretical Computer Science, 2012.

Definitions

Given a graph $G = (V, E)$ (strongly) connected.

Definition (Distance)

The distance $d(u, v)$ is the number (sum of the weights) of edges along shortest path from u to v.

Definition (Diameter)

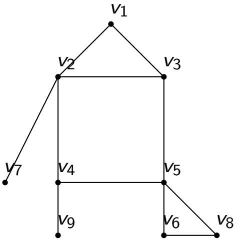
$$D = \max_{u,v \in V} d(u, v)$$

An Example Undirected Graph

Definition

- The eccentricity of a node u, $\mathrm{ecc}(u) = \max_{v \in V} d(u, v)$: in how many hops u can reach any node?

$$D = \max_{u \in V} \mathrm{ecc}(u)$$



	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	ecc
v_1	0	1	1	2	2	3	2	3	3	3
v_2	1	0	1	1	2	3	1	3	2	3
v_3	1	1	0	2	1	2	2	2	3	3
v_4	2	1	2	0	1	2	2	2	1	2
v_5	2	2	1	1	0	1	3	1	2	3
v_6	3	3	2	2	1	0	4	1	3	4
v_7	2	1	2	2	3	4	0	4	3	4
v_8	3	3	2	2	1	1	4	0	3	4
v_9	3	2	3	1	2	3	3	3	0	3

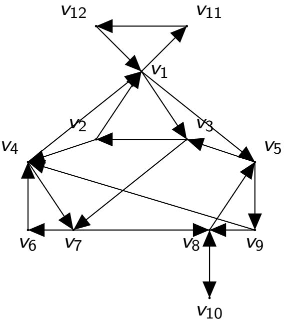
BFS [O(m) time]

For any i, $F_i(u)$ are the nodes at distance i from u (and vice versa).

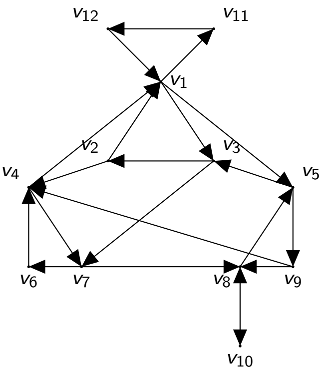
An Example Directed Graph

Definition

- Forward Eccentricity of u: in how many hops u can reach any node?
  - $\mathrm{ecc_F}(u) = \max_{v \in V} d(u, v)$
- Backward Eccentricity of u: in how many hops u can be reached starting from any node?
  - $\mathrm{ecc_B}(u) = \max_{v \in V} d(v, u)$
- Diameter: maximum $\mathrm{ecc_F}$ or $\mathrm{ecc_B}$



	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	ecc_F
v_1	0	2	1	3	1	3	2	3	2	4	1	2	4
v_2	1	0	2	1	2	3	2	3	3	4	2	3	4
v_3	2	1	0	2	3	2	1	2	4	3	3	4	4
v_4	1	3	2	0	2	2	1	2	3	3	2	3	3
v_5	3	2	1	2	0	3	2	2	1	3	4	5	5
v_6	2	4	3	1	3	0	2	3	4	4	3	4	4
v_7	3	4	3	2	2	1	0	1	3	2	4	5	5
v_8	4	3	2	3	1	4	3	0	2	1	5	6	6
v_9	2	4	3	1	2	3	2	1	0	2	3	4	4
v_{10}	5	4	3	4	2	5	4	1	3	0	6	7	7
v_{11}	2	4	3	5	3	5	4	5	4	6	0	1	6
v_{12}	1	3	2	4	2	4	3	4	3	5	2	0	5
ecc_B	5	4	3	5	3	5	4	5	4	6	6	7	



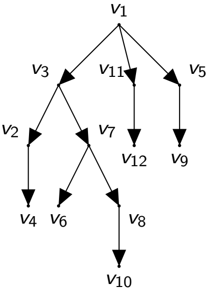
Forward BFS Tree [O(m) time]

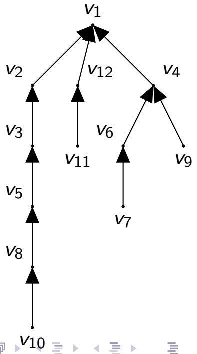
For any i, the forward fringe, $F_i^F(u)$ (which nodes are at distance i from u)

Backward BFS Tree [O(m) time]

For any i, the backward fringe, $F_i^B(u)$ (from which nodes, u is at distance i).

i	$F_i^F(v_1)$	$F_i^B(v_1)$
1	v_3, v_5, v_{11}	v_2, v_4, v_{12}
2	v_2, v_7, v_9, v_{12}	v_3, v_6, v_9, v_{11}
3	v_4, v_6, v_8	v_5, v_7
4	v_{10}	v_8
5		v_{10}





Motivations

Communication: completion time of broadcast protocols based on flooding

Social: how quickly information reaches every individual

Web: how quickly, in terms of mouse clicks, any page can be reached

State of the Art

- Textbook Algorithm ($n = |V|$, $m = |E|$). Too expensive.
  - Perform n BFS and return maximum ecc.
    - A BFS from x returns all the distances from x and takes $O(m)$ time.
- Several other approaches (see [Zwick, 2001]) that solves all pairs shortest path. Still too expensive.
  - $O(n^{(3+\omega)/2} \log n)$ where ω is the exponent of the matrix multiplication.
- Empirically finding lower bound L and upper bound U
  - That is, $L \leq D \leq U$
  - D found, when $L = U$

State of the Art: Negative Results

- Unless the so-called Strong Exponential Time Hypothesis (SETH) is false, deciding whether a graph has diameter 2 or 3 requires $\Omega(n^2)$.
  - Informally, SETH says that SAT cannot be solved in sub-exponential time.
- By this reduction, unless SETH fails, $\Omega(n^2)$ time is required to get a $(3/2 - \epsilon)$-approximation algorithm for computing the diameter even in the case of sparse graphs.

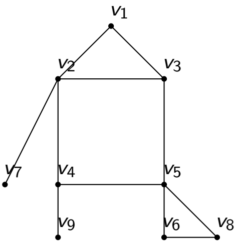
Liam Roditty, Virginia Vassilevska Williams: Fast approximation algorithms for the diameter and radius of sparse graphs. STOC 2013: 515-524

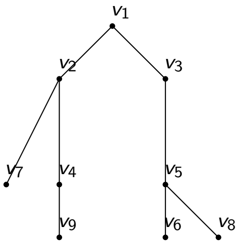
Part I

Computing lower and upper bound

Computing lower and upper bounds (undirected graph)

By using Single source (BFS) Shortest Path





Lower bound The eccentricity, ecc (height of the BFS tree) of a node.
In the example 3: at least a pair is at distance 3.

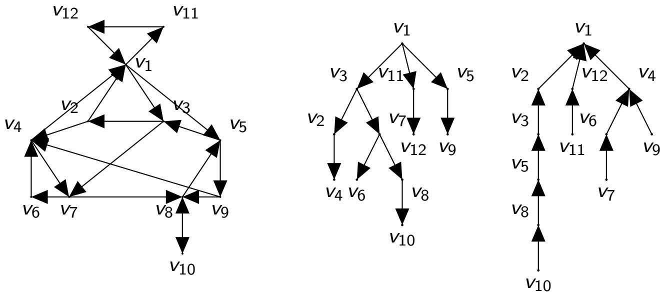
Upper bound The double of the eccentricity ecc of a node.
In the example 6: every node can reach another node going to v_1 by ≤ 3 edges and going to the destination in ≤ 3 edges.

$$x : d(x, u) = i \text{ and } y : d(u, y) = j \implies d(x, y) \leq i + j$$
$i + j$ is the length of a path from x to y passing through u.

- Bounds by sampling but very often $L < D < U$ (see SNAP experiments)
  In the example diameter is 4: $d(v_7, v_8) = 4$.

Computing lower and upper bounds (directed graph)

By using Single source (FBFS) and Single target (BBFS) Shortest Path



Lower Bound The forward eccentricity, ecc_F (height of the FBFS tree) of a node.
In the example 4: at least a pair is at distance 4.

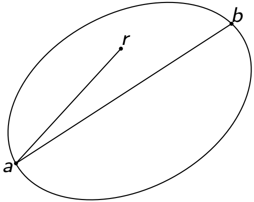
Upper Bound The forward eccentricity plus the backward eccentricity ecc_B (height of the BBFS tree) of a node.
In the example 9: every node can reach another node going to v_1 in ≤ 5 steps and going to the destination in ≤ 4 steps.

- Very often: $L < D < U$ (see SNAP experiments)
  In the example diameter is 7: $d(v_{10}, v_{12}) = 7$.

Good lower bounds in undirected graphs: 2-Sweep

2-Sweep

1. Run a BFS from a random node r: let a be the farthest node.
2. Run a BFS from a: let b be the farthest node.
3. Return the length of the path from a to b.



Return $d(a, b)$.

Experiments: effectiveness of 2-Sweep

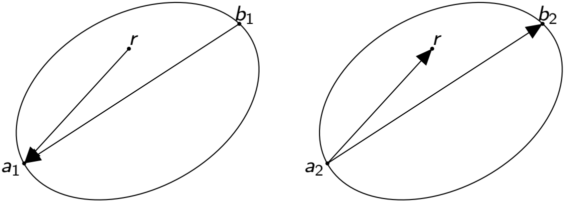
By starting from the highest degree node.

Category	# of Networks	2-dSweep*HdOut*	
		# of Networks in which *lb* is tight	Maximum error
Protein-Protein Interaction	14	11	1
Collaboration	14	12	1
Undirected Social	4	4	0
Undirected Communication	36	34	2
Autonomous System	2	1	1
Road	3	1	14
Word Adjacency	7	4	1

Better lower bounds in directed graphs: directed 2-Sweep

2-dSweep

1. Run a forward BFS from a random node r: let a_1 be the farthest node.
2. Run a backward BFS from a_1: let b_1 be the farthest node.
3. Run a backward BFS from r: let a_2 be the farthest node.
4. Run a forward BFS from a_2: let b_2 be the farthest node.
5. If $\mathrm{ecc}_B(a_1) > \mathrm{ecc}_F(a_2)$, then return the length of the path from b_1 to a_1. Otherwise return the length of the path from a_2 to b_2.



Return the maximum between $d(a_2, b_2)$ and $d(b_1, a_1)$.

First time used in directed graph by Broder et al. to study *Graph structure in the web*.

Experiments: effectiveness of 2-*d*SWEEP

By starting from the highest out-degree or the highest in-degree node.

Category	# of Networks	2-*d*SWEEP *HdOut*		2-*d*SWEEP *HdIn*	
		# of Networks in which *lb* is tight	Max error	# of Networks in which *lb* is tight	Max error
METABOLIC BIPARTITE	76	73	19	75	19
METABOLIC COMPOUND	76	73	9	75	9
METABOLIC REACTION	76	73	10	75	10
DIRECTED SOCIAL	10	10	0	10	0
WEB	16	16	0	16	0
CITATION	2	2	0	2	0
COMMUNICATION	3	3	0	2	1
P2P	9	8	1	7	1
PRODUCT CO-PURCHASING	5	5	0	5	0
WORD-ASSOCIATION	1	1	0	1	0

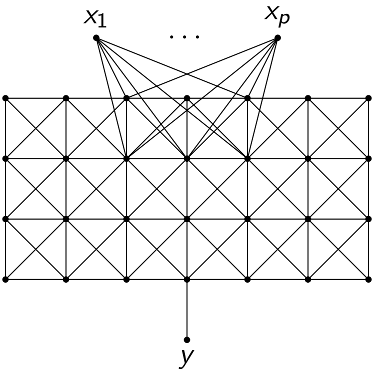
More experiments

Network name	D	Numb. of runs (out of 10) in which $LB = D$	Worst LB found
Wiki-Vote	9	10	9
p2p-Gnutella08	19	9	**18**
p2p-Gnutella09	19	9	**18**
p2p-Gnutella06	19	10	19
p2p-Gnutella05	22	9	**21**
p2p-Gnutella04	25	7	**22**
p2p-Gnutella25	21	8	**20**
p2p-Gnutella24	28	10	28
p2p-Gnutella30	23	2	**22**
p2p-Gnutella31	30	9	**29**
s.s.Slashdot081106	15	10	15
s.s.Slashdot090216	15	10	15
s.s.Slashdot090221	15	10	15
soc-Epinions1	16	9	**15**
Email-EuAll	10	10	10
soc-sign-epinions	16	10	16
web-NotreDame	93	10	93
Slashdot0811	12	10	12
Slashdot0902	13	3	**12**
WikiTalk	10	9	9
web-Stanford	210	10	210
web-BerkStan	679	10	679
web-Google	51	10	51

Network name	D	Numb. of runs (out of 10) in which $LB = D$	Worst LB found
wordassociation-2011	10	9	**9**
enron	10	10	10
uk-2007-05@100000	7	10	7
cnr-2000	81	10	81
uk-2007-05@1000000	40	10	40
in-2004	56	10	56
amazon-2008	47	10	47
eu-2005	82	10	82
indochina-2004	235	10	235
uk-2002	218	10	218
arabic-2005	133	10	133
uk-2005	166	10	166
it-2004	873	10	873

(`snap.stanford.edu` and `webgraph.dsi.unimi.it` dataset)

Bad cases for 2-SWEEP



In this modified grid with k rows and $1 + 3k/2$ columns. The algorithm can return k. The diameter of the network is instead $3k/2$.

Better lower bounds in undirected graphs: 4-SWEEP

- 4-SWEEP: apply 2-SWEEP pick the middle vertex of the path returned. Apply again 2-SWEEP.

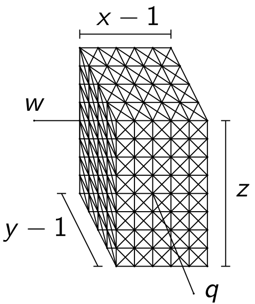
Algorithm 1: 4-SWEEP

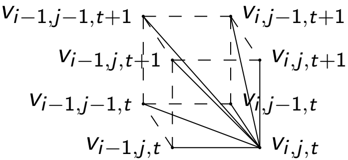
Input: A graph G

Output: A lower bound for the diameter of G and a node with (hopefully) low eccentricity

$r_1 \leftarrow$ random node of G or node with the highest degree;
$a_1 \leftarrow argmax_{v \in V} d(r_1, v)$;
$b_1 \leftarrow argmax_{v \in V} d(a_1, v)$;
$r_2 \leftarrow$ the node in the middle of the path between a_1 and b_1;
$a_2 \leftarrow argmax_{v \in V} d(r_2, v)$;
$b_2 \leftarrow argmax_{v \in V} d(a_2, v)$;
$u \leftarrow$ the node in the middle of the path between a_2 and b_2;
$lowerb \leftarrow \max\{\mathrm{ecc}(a_1), \mathrm{ecc}(a_2)\}$;
return $lowerb$ and u;

A bad case for the 4-SWEEP algorithm





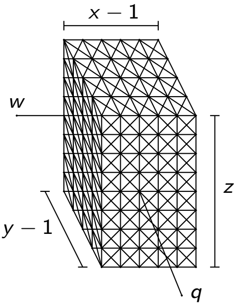
- For any two distinct nodes $v_{i,j,t}$ and $v_{i',j',t'}$,

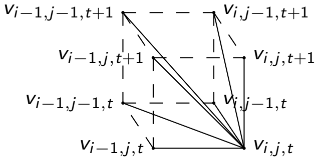
$$d(v_{i,j,t}, v_{i',j',t'}) = \max\{|i - i'|, |j - j'|, |t - t'|\}$$

$$\leq \max\{x - 2, y - 2, z - 1\}.$$

- The diameter of G is $\max\{x - 1, y, z - 1\}$.

If x, y, and z are such that $z > x > y + 1 > 3$, then the diameter is equal to $z - 1$. If $y - 1 > (z+1)/2$, $y - 1 > (x+1)/2$, and $x - 1 > (z+1)/2$, then the lower bound computed by the 4-SWEEP algorithm can be $x - 1$ instead of $z - 1$. (approximation ratio asymptotically close to 2)





- $r_1 = v_{x-1,1,(z+1)/2}$.
- $a_1 = w$ since $x - 1 > (z+1)/2 - 1$ and $x - 1 > y - 1$.
- $b_1 = v_{x-1,1,(z+1)/2}$ since $x - 1 > (z+1)/2$ and $x - 1 > y$.
- $r_2 = v_{(x-1)/2,1,(z+1)/2}$, that is the node opposite to q with respect to the x axis.
- $a_2 = q$ since $y - 1 > (z+1)/2 - 1$ and $y - 1 > (x-1)/2$.
- $b_2 = w$, since $y > (z+1)/2$ and $y > (x+1)/2$.

Thus $\mathrm{ecc}(r_1) = \mathrm{ecc}(a_1) = x - 1$, $\mathrm{ecc}(r_2) = y - 1$, and $\mathrm{ecc}(a_2) = y$. The lower bound computed is $\max\{x-1, y-1, y\} = x - 1$.

Open problems

- Understand why 2-Sweep both in the directed and in the undirected version, is so effective in finding tight lower bounds.
- It has been proved that in chordal graph the error of the 2-Sweep can be at most 1.
- Which is the topological underlying property that can lead us to these results? Might be related to some sort of chordality measure? Why real world graphs exhibit this property?

- Clemence Magnien, Matthieu Latapy, Michel Habib: Fast computation of empirically tight bounds for the diameter of massive graphs. ACM Journal of Experimental Algorithmics 13 (2008)
- Pierluigi Crescenzi, Roberto Grossi, Claudio Imbrenda, Leonardo Lanzi, Andrea Marino: Finding the Diameter in Real-World Graphs - Experimentally Turning a Lower Bound into an Upper Bound. ESA (1) 2010: 302-313

Part II

Computing exactly the diameter

Bound Refinement: iterative fringe upper bound

Reminder

The *textbook* algorithm runs a BFS for any node and return the maximum ecc found.

Main Schema of a new algorithm

- Perform the BFSes one after the other specifying the order in which the BFSes have to be executed, while doing this:
  - refine a lower bound: that is the maximum ecc found until that moment.
  - upper bound the eccentricities of the remaining nodes.
  - stop when the remaining nodes cannot have eccentricity higher than our lower bound.

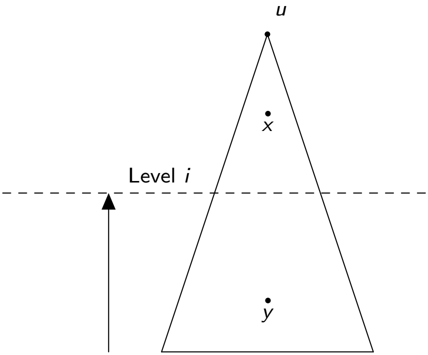
Good order can be inferred looking at some properties of BFS trees.

Notation

- Let u be any node in V and let us denote the set $\{v \mid d(u,v) = \mathrm{ecc}(u)\}$ of nodes at maximum distance $\mathrm{ecc}(u)$ from u as $F(u)$.
- Let $F_i(u)$ be the *fringe* set of nodes at distance i from u (note that $F(u) = F_{\mathrm{ecc}(u)}(u)$)
- Let $B_i(u) = \max_{z \in F_i(u)} \mathrm{ecc}(z)$ be the maximum eccentricity among these nodes.

Main observation

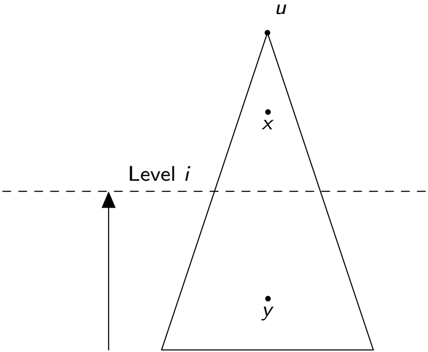
All the nodes x above the level i in $\mathrm{BFS}(u)$ having ecc greater than $2(i-1)$ have a corresponding node y, whose ecc is greater or equal to $\mathrm{ecc}(x)$, below or on the level i in $\mathrm{BFS}(u)$.



Main Theorem

Theorem

For any $1 \leq i < \mathrm{ecc}(u)$ *and* $1 \leq k < i$*, and for any* $x \in F_{i-k}(u)$ *such that* $\mathrm{ecc}(x) > 2(i-1)$*, there exists* $y \in F_j(u)$ *such that* $d(x, y) = \mathrm{ecc}(x)$ *with* $j \geq i$*.*



Proof: some observations

Observation

For any x *and* y *in* V *such that* $x \in F_i(u)$ *or* $y \in F_i(u)$*, we have that* $d(x, y) \leq B_i(u)$*.*

Indeed, $d(x, y) \leq \min\{\mathrm{ecc}(x), \mathrm{ecc}(y)\} \leq B_i(u)$.

Observation

For any $1 \leq i, j \leq \mathrm{ecc}(u)$ *and for any* $x \in F_i(u)$ *and* $y \in F_j(u)$*, we have* $d(x, y) \leq i + j \leq 2 \max\{i, j\}$*.*

Proof

Theorem

For any $1 \leq i < \mathrm{ecc}(u)$ *and* $1 \leq k < i$, *and for any* $x \in F_{i-k}(u)$ *such that* $\mathrm{ecc}(x) > 2(i-1)$, *there exists* $y_x \in F_j(u)$ *such that* $d(x, y_x) = \mathrm{ecc}(x)$ *with* $j \geq i$.

Proof.

Since $\mathrm{ecc}(x) > 2(i-1)$, then there exists y_x whose distance from x is equal to $\mathrm{ecc}(x)$ and, hence, greater than $2(i-1)$.
If y_x was in $F_j(u)$ with $j < i$, then from the previous observation it would follow that

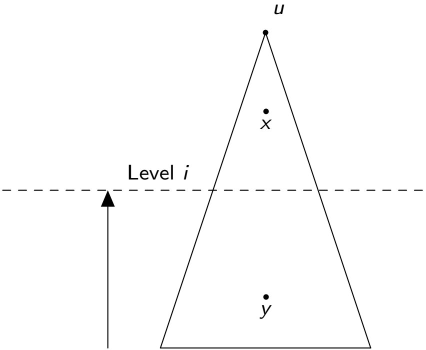
$$d(x, y_x) \leq 2\max\{i-k, j\} \leq 2\max\{i-k, i-1\} = 2(i-1),$$

which is a contradiction.
Hence, y_x must be in $F_j(u)$ with $j \geq i$. □

Implication

Corollary

Let lb be the maximum eccentricity among all the eccentricities of the nodes in or below the level i. The eccentricities of all the nodes above the level i is bounded by $\max\{lb, 2(i-1)\}$.



Implication

Corollary

Let lb be the maximum eccentricity among all the eccentricities of the nodes in or below the level i. The eccentricities of all the nodes above the level i is bounded by $\max\{lb, 2(i-1)\}$.

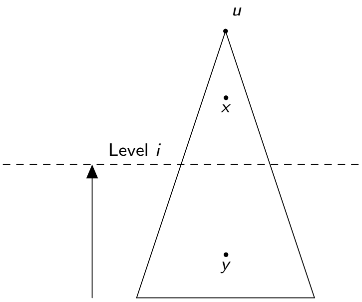
Proof.

For any node x above the level i, there are two cases:

- $ecc(x) \leq 2(i-1)$
- $ecc(x) > 2(i-1)$. The Theorem applies: there is a node y below the level i whose eccentricity $ecc(y)$ is greater than or equal to $ecc(x)$.

$$lb \geq ecc(y) \geq ecc(x)$$

□



Back to the main schema

Perform the BFSes one after the others following the order induced by the BFS tree of a node u: starting from the nodes in $F(u)$, go in a bottom-up fashion.

- At each level i compute the eccentricities of all its nodes: if the maximum eccentricity found lb is greater than $2(i-1)$ then we can discard traversing the remaining levels, since the eccentricities of all their nodes cannot be greater than lb.
  - Since the eccentricity of the remaining nodes is bounded by $\max\{lb, 2(i-1)\}$

Given a node u.

- Set $i = \mathrm{ecc}(u)$ and $M = B_i(u)$.
- If $M > 2(i-1)$, then return M; else, set $i = i-1$ and $M = \max\{M, B_i(u)\}$, and repeat this step.

Algorithm 2: *i*FUB

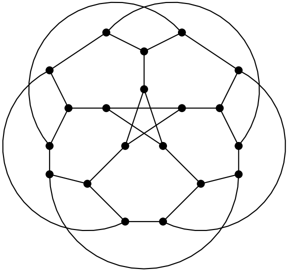
Input: A graph G, a node u
Output: The diameter D

$i \leftarrow \mathrm{ecc}(u)$;
$lb \leftarrow \mathrm{ecc}(u)$;
$ub \leftarrow 2\mathrm{ecc}(u)$;
while $ub > lb$ **do**
  if $\max\{lb, B_i(u)\} > 2(i-1)$ **then**
    return $\max\{lb, B_i(u)\}$;
  else
    $lb \leftarrow \max\{lb, B_i(u)\}$;
    $ub \leftarrow 2(i-1)$;
  end
  $i \leftarrow i - 1$;
end
return lb

Bad cases (1)

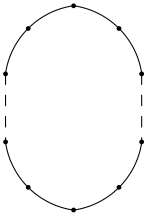
Two bad things can happen:

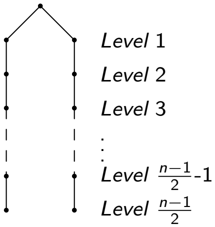
- The amount of nodes in $F(u)$ is linear: for instance if the BFS tree of the starting node is a binary tree.
  - In special regular graphs [such as Moore graphs] no good choice is possible.



Bad cases (2)

- Cases in which nodes have close eccentricity or the BFS trees are isomorphic, like the Moore graph, but even simpler: a cycle.
  - A cycle with n nodes (n odd) has diameter $\frac{n-1}{2}$, and each node has the same BFS tree.
  - The loop is repeated until $2(i-1) \geq \frac{n-1}{2}$, that is $i \geq \frac{n+3}{4}$, and stops the first time that $2(i-1) < \frac{n-1}{2}$.
  - The total number of iterations is equal to $\frac{n-1}{2} - \frac{n+3}{4} + 2 = \frac{n+3}{4}$.
  - The number of BFSes is $\frac{n+3}{2}$.

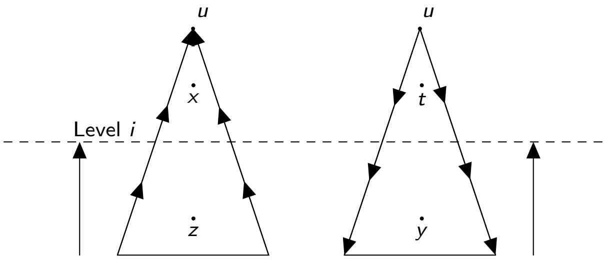




directed iterative fringe upper bound

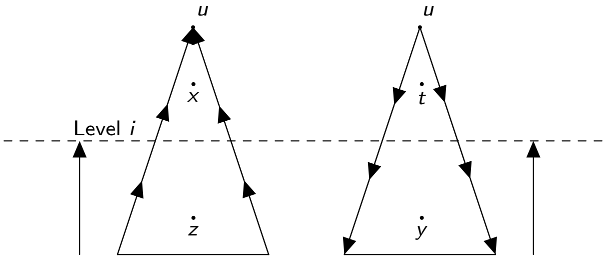
Theorem (Sketch)

1. *All the nodes x above the level i in* $\mathrm{BBFS}(u)$ *having* $\mathrm{ecc_F}$ *greater than* $2(i-1)$ *have a corresponding node y, whose* $\mathrm{ecc_B}$ *is greater or equal to* $\mathrm{ecc_F}(x)$*, below or on the level i in* $\mathrm{FBFS}(u)$*.*
2. *All the nodes t above the level i in* $\mathrm{FBFS}(u)$ *having* $\mathrm{ecc_B}$ *greater than* $2(i-1)$ *have a corresponding node z, whose* $\mathrm{ecc_F}$ *is greater or equal to* $\mathrm{ecc_B}(t)$*, below or on the level i in* $\mathrm{BBFS}(u)$*.*



Corollary

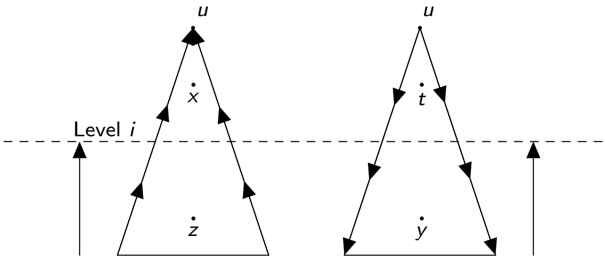
- *Let lb be the maximum among all the* ecc_B *of nodes y and among all the* ecc_F *of nodes z.*
- *The* ecc_B *of nodes t and the* ecc_F *of nodes x are bounded by* $\max\{lb, 2(i-1)\}$.



Back to the main schema

For decreasing values of i

- At level i we have computed all the $\mathrm{ecc_B}$ of nodes y and the $\mathrm{ecc_F}$ of nodes z. The maximum is our lower bound lb.
- If lb is bigger than $2(i-1)$, lb is the diameter.
  - No node to be examined can have $\mathrm{ecc_F}$ or $\mathrm{ecc_B}$ bigger than lb.



More formally

$$B_j^F(u) = \begin{cases} \max_{x \in F_j^F(u)} \mathrm{ecc}_B(x) & \text{if } j \leq \mathrm{ecc}_F(u), \\ 0 & \text{otherwise} \end{cases}$$

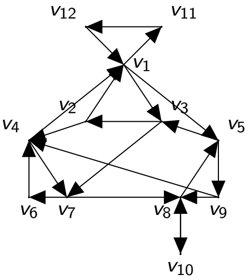
$$B_j^B(u) = \begin{cases} \max_{x \in F_j^B(u)} \mathrm{ecc}_F(x) & \text{if } j \leq \mathrm{ecc}_B(u), \\ 0 & \text{otherwise.} \end{cases}$$

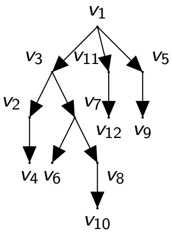
Algorithm 3: DiFUB

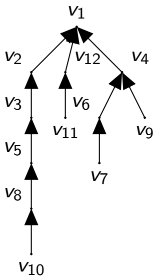
Input: A strongly connected di-graph G, a node u
Output: The diameter D

$i \leftarrow \max\{\mathrm{ecc}_F(u), ecc_B(u)\}$;
$lb \leftarrow \max\{\mathrm{ecc}_F(u), ecc_B(u)\}$;
$ub \leftarrow 2i$;
while $ub > lb$ **do**
    if $\max\{lb, B_i^B(u), B_i^F(u)\} > 2(i-1)$ **then**
        return $\max\{lb, B_i^B(u), B_i^F(u)\}$;
    else
        $lb \leftarrow \max\{lb, B_i^B(u), B_i^F(u)\}$;
        $ub \leftarrow 2(i-1)$;
    end
    $i \leftarrow i - 1$;
end
return lb;

Example



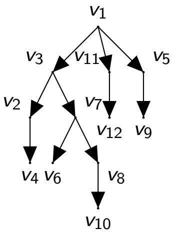


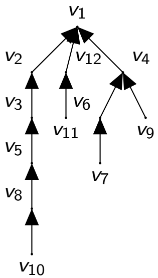


	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	ecc_F
v_1	0	2	1	3	1	3	2	3	2	4	1	2	4
v_2	1	0	2	1	2	3	2	3	3	4	2	3	4
v_3	2	1	0	2	3	2	1	2	4	3	3	4	4
v_4	1	3	2	0	2	2	1	2	3	3	2	3	3
v_5	3	2	1	2	0	3	2	2	1	3	4	5	5
v_6	2	4	3	1	3	0	2	3	4	4	3	4	4
v_7	3	4	3	2	2	1	0	1	3	2	4	5	5
v_8	4	3	2	3	1	4	3	0	2	1	5	6	6
v_9	2	4	3	1	2	3	2	1	0	2	3	4	4
v_{10}	5	4	3	4	2	5	4	1	3	0	6	7	7
v_{11}	2	4	3	5	3	5	4	5	4	6	0	1	6
v_{12}	1	3	2	4	2	4	3	4	3	5	2	0	5
ecc_B	5	4	3	5	3	5	4	5	4	6	6	7	

If we choose $u = v_1$, the corresponding two breadth-first search trees are T_u^F and T_u^B.

i	$F_i^F(v_1)$	$F_i^B(v_1)$
1	v_3, v_5, v_{11}	v_2, v_4, v_{12}
2	v_2, v_7, v_9, v_{12}	v_3, v_6, v_9, v_{11}
3	v_4, v_6, v_8	v_5, v_7
4	v_{10}	v_8
5		v_{10}



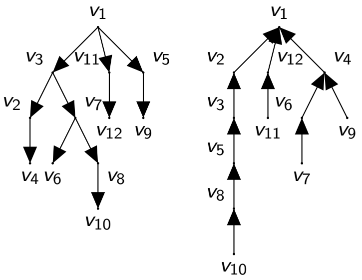


The main theorem says for instance:
if we choose $i = 2$, $k = 1$, and $x = v_4 \in F_1^B(v_1)$, then we have that $\mathrm{ecc}_F(v_4) = 3 > 2 = 2(i-1)$: the theorem is witnessed by node $y = v_2 \in F_2^F(v_1)$ (indeed, $d(v_4, v_2) = 3$).

Algorithm 4: DiFUB

Input: A strongly connected di-graph G, a node u
Output: The diameter D

$i \leftarrow \max\{\mathrm{ecc}_F(u), ecc_B(u)\}$;
$lb \leftarrow \max\{\mathrm{ecc}_F(u), ecc_B(u)\}$;
$ub \leftarrow 2i$;
while $ub > lb$ **do**
  if $\max\{lb, B_i^B(u), B_i^F(u)\} > 2(i-1)$ **then**
    return $\max\{lb, B_i^B(u), B_i^F(u)\}$;
  else
    $lb \leftarrow \max\{lb, B_i^B(u), B_i^F(u)\}$;
    $ub \leftarrow 2(i-1)$;
  end
  $i \leftarrow i - 1$;
end
return lb;



Invoke the algorithm with $u = v_1$.

1. $i = lb = \max\{\mathrm{ecc}_F(v_1), \mathrm{ecc}_B(v_1)\} = \max\{4, 5\} = 5$, and $ub = 2i = 10$.
2. Since $ub - lb = 5 > 0$, the algorithm enters the **while** loop with $i = 5$.
3. Since $5 > \mathrm{ecc}_F(u)$, $B_5^F(u) = 0$, and $B_5^B(u) = \mathrm{ecc}_F(v_{10}) = 7$: since, $7 < 8 = 2(i-1)$, the algorithm enters the **else** branch and set lb equal to 7 and ub equal to 8.
4. $ub - lb = 1 > 0$ and we enter again into the **while** loop with $i = 4$.
5. $B_4^F(u) = \mathrm{ecc}_B(v_{10}) = 6$ and $B_4^B(u) = \mathrm{ecc}_F(v_8) = 6$: hence, $\max\{lb, B_4^B(u), B_4^F(u)\} = 7 > 6 = 2(i-1)$.
6. The algorithm enters the **if** branch and returns the value 7 which is the correct diameter value.

Bad cases

Same as for undirected graphs, replacing each edge by two opposite arcs.

The choice of the starting node

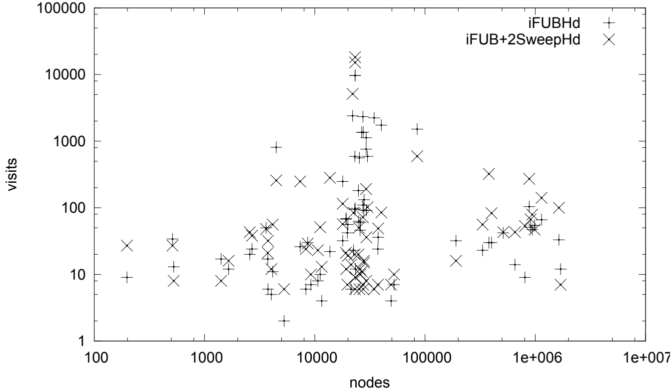
It heavily affects the performance (number of visits we will do).

Try to use a vertex with low eccentricity (i.e., central, well connected).

- High degree node (in-degree or out-degree).
- The node in the middle of the diametral path returned by 2-SWEEP.

Experiments for undirected graphs

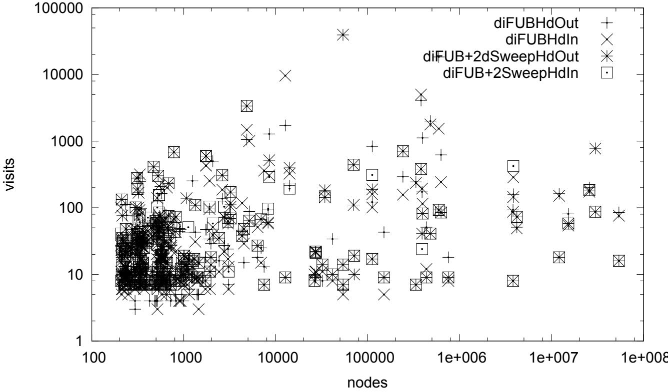
The number of visits seems to be constant.



It has been used to compute the diameter of Facebook (721.1M nodes and 68.7G edges, Diameter 41) with just 17 BFSes.

Experiments for directed graphs

The number of visits seems to be constant.



More Experiments

Network name	n	m	Avg. Visits	Visits worst run
Wiki-Vote	1300	39456	17	17
p2p-Gnutella08	2068	9313	45.9	64
p2p-Gnutella09	2624	10776	202.1	230
p2p-Gnutella06	3226	13589	236.6	279
p2p-Gnutella05	3234	13453	60.4	94
p2p-Gnutella04	4317	18742	36.7	38
p2p-Gnutella25	5153	17695	85.1	161
p2p-Gnutella24	6352	22928	13	13
p2p-Gnutella30	8490	31706	255.4	516
p2p-Gnutella31	14149	50916	208.7	255
s.s.Slashdot081106	26996	337351	22.3	25
s.s.Slashdot090216	27222	342747	21.5	26
s.s.Slashdot090221	27382	346652	22.8	26
soc-Epinions1	32223	443506	6.1	7
Email-EuAll	34203	151930	6	6
soc-sign-epinions	41441	693737	6	6
web-NotreDame	53968	304685	7	7
Slashdot0811	70355	888662	40	40
Slashdot0902	71307	912381	32.9	40
WikiTalk	111881	1477893	13.6	19
web-Stanford	150532	1576314	6	6
web-BerkStan	334857	4523232	7	7
web-Google	434818	3419124	9.4	10

(`snap.stanford.edu` dataset)

More Experiments

Network name	n	m	Avg. Visits	Visits worst run
wordassociation-2011	4845	61567	412.5	423
enron	8271	147353	19	22
uk-2007-05@100000	53856	1683102	14	14
cnr-2000	112023	1646332	17	17
uk-2007-05@1000000	480913	22057738	6	6
in-2004	593687	7827263	14	14
amazon-2008	627646	4706251	136.3	598
eu-2005	752725	17933415	6	6
indochina-2004	3806327	98815195	8	8
uk-2002	12090163	232137936	6	6
arabic-2005	15177163	473619298	58	58
uk-2005	25711307	704151756	170	170
it-2004	29855421	938694394	87	87

(`webgraph.dsi.unimi.it` dataset)

The number of visits seems to be constant.
$\Rightarrow$ For any graph with more than 10000 nodes, DiFUB performs $0.001\% n$ visits in stead of n.

Why so well?

Suitable properties of the starting node u

(1) u has to be the node with minimum eccentricity, called radius R.

(2) Constant number of nodes in $F(u)$.

- If you are able to infer the node u such that (1) and $R = D/2$ you will stop after one iteration.
  - High degree node is very often a good choice.
  - If the lower bound path returned by 2-Sweep is tight and $R = D/2$, the node in the middle of this path make us stop after one iteration.
- Almost always in real-world graphs $R = D/2$ (the minimum possible, maximum heterogeneity) and (2) is true if u is central.

Further development

- DiFUB can be generalized to weighted graphs, using Dijkstra Algorithm instead of BFS and sorting the nodes according to their distance from u. It works well, but not for Road Networks.
- Further optimization allow to do better than this and to compute also the diameter of weakly connected graphs (Borassi et al., TCS 2015).
- It is possible to prove that for the configuration model fixing the power law the number of BFSes is almost constant.

Open questions

- Why the *i*FUB method works so well in general.
  - Might be related to eccentricities distribution?

- Pierluigi Crescenzi, Roberto Grossi, Leonardo Lanzi, Andrea Marino: On Computing the Diameter of Real-World Directed (Weighted) Graphs. SEA 2012: 99-110
- Pilu Crescenzi, Roberto Grossi, Michel Habib, Leonardo Lanzi, Andrea Marino: On computing the diameter of real-world undirected graphs. Theor. Comput. Sci. 514: 84-95 (2013)
- Frank W. Takes, Walter A. Kosters: Determining the diameter of small world networks. CIKM 2011: 1191-1196
- Michele Borassi, Pierluigi Crescenzi, Michel Habib, Walter A. Kosters, Andrea Marino, Frank W. Takes: On the Solvability of the Six Degrees of Kevin Bacon Game - A Faster Graph Diameter and Radius Computation Method. FUN 2014: 52-63

Thanks