Diameter computations in graphs, from practice to theory back and forth

Michel Habib
Prof. emeritus
IRIF Paris University & Gang, Inria project Paris
habib@irif.fr
http://www.irif.fr/~habib

Hyderabad
10 février 2020

Schedule of this talk

Joint work with :
D. Corneil (Toronto), C. Paul (Montpellier), F. Dragan (Kent), V. Chepoi (Marseille), B. Estrellon (Marseille), Y. Vaxes (Marseille), Y. Xiang (Kent), C. Magnien (Paris), M. Latapy (Paris), P. Crescenzi (Firenze), R. Grossi (Pisa), A. Marino (Pisa), M. Borassi (Firence), W. Kosters, F. Takes, Laurent Viennot (Irif Paris), H. Alrasheed (Kent), G. Ducoffe (Bucarest)
and many discussions with others . . .

Basics Definitions

Definitions :

Let G be an undirected graph :

- $exc(x) = max_{y \in G}\{distance(x, y)\}$ excentricity
- $diam(G) = max_{x \in G}\{exc(x)\}$ diameter
- $radius(G) = min_{x \in G}\{exc(x)\}$
- $x \in V$ is a center of G, if $exc(x) = radius(G)$

Basics Definitions

Definitions :

Let G be an undirected graph :

- $exc(x) = max_{y \in G}\{distance(x, y)\}$ excentricity
- $diam(G) = max_{x \in G}\{exc(x)\}$ diameter
- $radius(G) = min_{x \in G}\{exc(x)\}$
- $x \in V$ is a center of G, if $exc(x) = radius(G)$

First remarks of the definitions

distance computed in # edges
If x and y belong to different connected components $d(x, y) = \infty$.
diameter : Max Max Min
radius : Min Max Min

Trivial bounds

For any graph G :
$radius(G) \leq diam(G) \leq 2radius(G)$ and $\forall e \in G$,
$diam(G) \leq diam(G - e)$

Trivial bounds

For any graph G :
$radius(G) \leq diam(G) \leq 2radius(G)$ and $\forall e \in G$,
$diam(G) \leq diam(G - e)$

These bounds are tight

Trivial bounds

For any graph G :
$radius(G) \leq diam(G) \leq 2radius(G)$ and $\forall e \in G$,
$diam(G) \leq diam(G-e)$

These bounds are tight

- If G is a path of length 2K, then $diam(G) = 2k = 2radius(G)$, and G admits a unique center, i.e. the middle of the path.

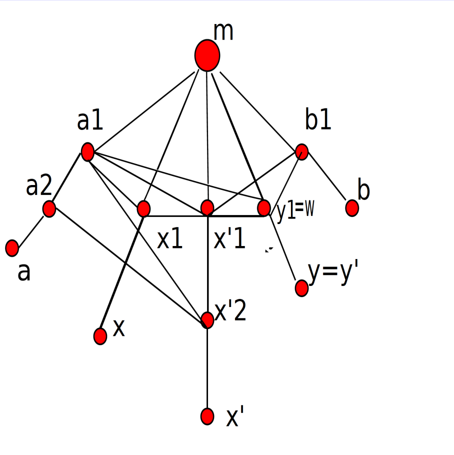
Trivial bounds

For any graph G :
$radius(G) \leq diam(G) \leq 2radius(G)$ and $\forall e \in G$,
$diam(G) \leq diam(G - e)$

These bounds are tight

- If G is a path of length 2K, then $diam(G) = 2k = 2radius(G)$, and G admits a unique center, i.e. the middle of the path.
- If $radius(G) = diam(G)$, then $Center(G) = V$. All vertices are centers (as for example in a cycle).

First example $diam(G) = 5 = length[a, b]$



If $2.radius(G) = diam(G)$, then *roughly* G has a tree shape (at least it works for trees).
But there is no nice characterization of this class of graphs.

Diameter

Applications

1. A graph parameter which measures the quality of services of a network, in terms of worst cases, when all have a unitary cost. Find critical edges e s.t. $diam(G-e) > diam(G)$

Diameter

Applications

1. A graph parameter which measures the quality of services of a network, in terms of worst cases, when all have a unitary cost. Find critical edges e s.t. $diam(G - e) > diam(G)$
2. Many distributed algorithms can be analyzed with this parameter (when a flooding technique is used to spread information over the network or to construct routing tables).

Diameter

Applications

1. A graph parameter which measures the quality of services of a network, in terms of worst cases, when all have a unitary cost. Find critical edges e s.t. $diam(G - e) > diam(G)$
2. Many distributed algorithms can be analyzed with this parameter (when a flooding technique is used to spread information over the network or to construct routing tables).
3. Verify the small world hypothesis in some large social networks, using J. Kleinberg's definition of small world graphs.

Diameter

Applications

1. A graph parameter which measures the quality of services of a network, in terms of worst cases, when all have a unitary cost. Find critical edges e s.t. $diam(G - e) > diam(G)$
2. Many distributed algorithms can be analyzed with this parameter (when a flooding technique is used to spread information over the network or to construct routing tables).
3. Verify the small world hypothesis in some large social networks, using J. Kleinberg's definition of small world graphs.
4. Compute the diameter of the Internet graph, or some Web graphs, i.e. massive data.

1. Examples of diameter searches based on the algorithms presented in this course :
   `https://files.inria.fr/gang/road/`

1. Examples of diameter searches based on the algorithms presented in this course : `https://files.inria.fr/gang/road/`
2. OpenStreetMap (OSM) : 80 millions of nodes, average degree 3

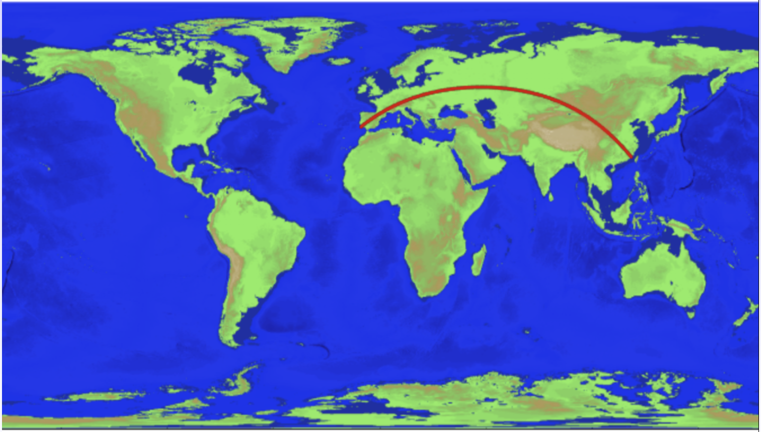
1. Examples of diameter searches based on the algorithms presented in this course :
   `https://files.inria.fr/gang/road/`
2. OpenStreetMap (OSM) : 80 millions of nodes, average degree 3
3. Roadmaps graphs a special domain of research interest
   Quasi-planar graph (bridges on the roads)

1. Examples of diameter searches based on the algorithms presented in this course : `https://files.inria.fr/gang/road/`
2. OpenStreetMap (OSM) : 80 millions of nodes, average degree 3
3. Roadmaps graphs a special domain of research interest Quasi-planar graph (bridges on the roads)
4. Never forget that computer science has an important experimental part.

1. Examples of diameter searches based on the algorithms presented in this course : `https://files.inria.fr/gang/road/`
2. OpenStreetMap (OSM) : 80 millions of nodes, average degree 3
3. Roadmaps graphs a special domain of research interest Quasi-planar graph (bridges on the roads)
4. Never forget that computer science has an important experimental part.
5. Many algorithmic ideas come out some experiment.

Longest straight line on land

[Chabukswar, Mukherjee 2018] :



"The problem was approached as a purely mathematical exercise. The authors do not recommend sailing or driving along the found paths."

Longest straight line on water

Reddit user kepleronlyknows 2012 (in r/MapPorn) :

Frequently Asked Questions (FAQ)

Usual questions on diameter, centers and radius :

- What is the best Program (resp. algorithm) available ?

Frequently Asked Questions (FAQ)

Usual questions on diameter, centers and radius :

- What is the best Program (resp. algorithm) available ?
- What is the complexity of diameter, center and radius computations ?

Frequently Asked Questions (FAQ)

Usual questions on diameter, centers and radius :

- What is the best Program (resp. algorithm) available ?
- What is the complexity of diameter, center and radius computations ?
- How to compute or approximate the diameter of huge graphs ?

Frequently Asked Questions (FAQ)

Usual questions on diameter, centers and radius :

- What is the best Program (resp. algorithm) available ?
- What is the complexity of diameter, center and radius computations ?
- How to compute or approximate the diameter of huge graphs ?
- Find a center (or all centers) in a network, (in order to install serveurs).

Some notes

1. I was asked first this problem in 1980 by France Telecom for the phone network (FT granted a PhD).

Some notes

1. I was asked first this problem in 1980 by France Telecom for the phone network (FT granted a PhD).
2. Marc Lesk obtained his PhD in 1984 with the title :
   Couplages maximaux et diamètres de graphes.
   Maximum matchings and diameter computations

Some notes

1. I was asked first this problem in 1980 by France Telecom for the phone network (FT granted a PhD).
2. Marc Lesk obtained his PhD in 1984 with the title :
   Couplages maximaux et diamètres de graphes.
   Maximum matchings and diameter computations
3. But, with very little practical results for diameter computations. (Some remarks like : If $diam(\overline{G}) \geq 6$ then $diam(G) \leq 2$, but with no algorithmic use)

- Our aim is to design an algorithm or heuristic to compute the diameter of very large graphs

- Our aim is to design an algorithm or heuristic to compute the diameter of very large graphs
- Any algorithm that computes all distances between all pairs of vertices, complexity $O(n^3)$ or $O(nm)$. As for example with $|V|$ successive Breadth First Searches in $O(n(n+m))$.

- Our aim is to design an algorithm or heuristic to compute the diameter of very large graphs
- Any algorithm that computes all distances between all pairs of vertices, complexity $O(n^3)$ or $O(nm)$. As for example with $|V|$ successive Breadth First Searches in $O(n(n+m))$.
- Best known complexity for an exact algorithm is $O(\frac{n^3}{log^3 n})$, in fact computing all shortest paths.

- Our aim is to design an algorithm or heuristic to compute the diameter of very large graphs
- Any algorithm that computes all distances between all pairs of vertices, complexity $O(n^3)$ or $O(nm)$. As for example with $|V|$ successive Breadth First Searches in $O(n(n+m))$.
- Best known complexity for an exact algorithm is $O(\frac{n^3}{log^3 n})$, in fact computing all shortest paths.
- But also with at most $O(Diam(G))$ matrix multiplications.

- Clemence Magnien and M. Latapy asked me again (2006) this question about diameter.

- Clemence Magnien and M. Latapy asked me again (2006) this question about diameter.
- But in the meantime, I met Derek Corneil and Feodor Dragan, we proved some theorems about diameter and chordals graphs but above all I had learned many properties of graph searches from Derek Corneil.

- Clemence Magnien and M. Latapy asked me again (2006) this question about diameter.
- But in the meantime, I met Derek Corneil and Feodor Dragan, we proved some theorems about diameter and chordals graphs but above all I had learned many properties of graph searches from Derek Corneil.
- I answered to Olivier Gascuel's usual question, how to compute diameter of phylogenetic trees, using the following algorithm.

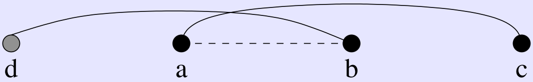
BFS

Data: a graph $G = (V, E)$ and a start vertex $s \in V$
Result: an ordering σ of V
Initialize *queue* to $\{s\}$
for $i \leftarrow 1$ **à** n **do**
    dequeue v from beginning of *queue*
    $\sigma(i) \leftarrow v$
    foreach *unnumbered vertex* $w \in N(v)$ **do**
        if w *is not already in queue* **then**
            enqueue w to the end of *queue*
        end
    end
end

Algorithm 1: Breadth First Search (BFS)

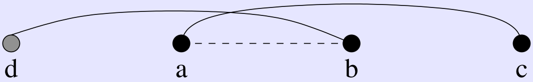
Property (B)

For an ordering σ on V, if for every triple (a, b, c), such that $a <_\sigma b <_\sigma c$ and $ac \in E$ and $ab \notin E$, then there must exist a vertex d such that $d <_\sigma a$ with $db \in E$



Property (B)

For an ordering σ on V, if for every triple (a, b, c), such that $a <_\sigma b <_\sigma c$ and $ac \in E$ and $ab \notin E$, then there must exist a vertex d such that $d <_\sigma a$ with $db \in E$



Theorem Corneil and Krueger 2008

For a graph $G = (V, E)$, an ordering σ sur V is a BFS ordering of G iff σ satisfies property (B).

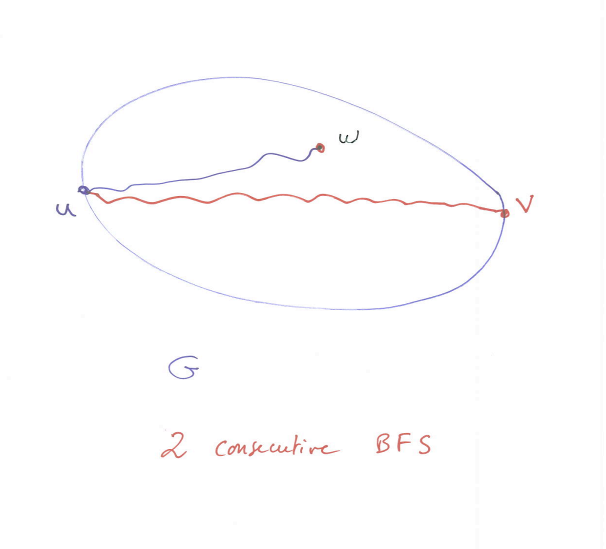
1. Let us consider the procedure called : 2 consecutive BFS [1]

Data: A graph $G = (V, E)$
Result: u, v two vertices
Choose a vertex $w \in V$
$u \leftarrow BFS(w)$
$v \leftarrow BFS(u)$

Where BFS stands for Breadth First Search.
Therefore it is a linear procedure

1. Proposed the first time by Handler 1973

Intuition behind the procedure



- Handler's classical result 73
  If G is a tree, $diam(G) = d(u, v)$
  Easy using Jordan's theorem.

First theorem

Camille Jordan 1869 :

A tree admits one or two centers depending on the parity of its diameter and furthermore all chains of maximum length starting at any vertex contain this (resp. these) centers.
And $radius(G) = \lceil \frac{diam(G)}{2} \rceil$

First theorem

Camille Jordan 1869 :

A tree admits one or two centers depending on the parity of its diameter and furthermore all chains of maximum length starting at any vertex contain this (resp. these) centers.
And $radius(G) = \lceil \frac{diam(G)}{2} \rceil$

- Camille Jordan, *Sur les assemblages de lignes*, Journal für reine und angewandte Mathematik **70** (1869), 185–190.

Unfortunately it is not an algorithm !

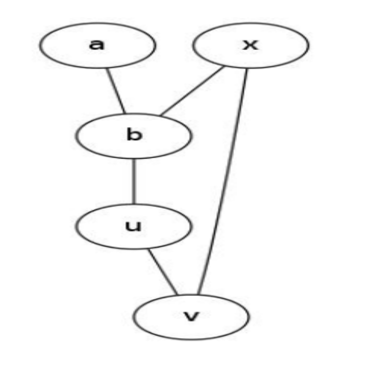


FIGURE: $BFS1 = x, b, v, a, u$, and $BFS2 = u, b, v, x, a$. But $diam(G) = 3$ with $[a, b, u, v]$, but starting from b the procedure finds the diameter.

Since the 2-sweep procedure does not give always the right answer, from every starting vertex. It is not an algorithm, just an heuristics.

Certificates for the diameter

To give a certificate $diam(G) = k$, it is enough to provide :

- two vertices x, y s.t. $d(x, y) = k$ ($diam(G) \geq k$).

Certificates for the diameter

To give a certificate $diam(G) = k$, it is enough to provide :

- two vertices x, y s.t. $d(x, y) = k$ ($diam(G) \geq k$).
- a subgraph $H \subset G$ with $diam(H) = k$ ($diam(G) \leq k$).
  H may belong to a class of graphs on which diameter computations can be done in linear time, for example trees.

Experimental results : M.H., M.Latapy, C. Magnien 2008

Randomized BFS procedure

Data: A graph $G = (V, E)$
Result: u, v two vertices
Repeat α times :
Randomly Choose a vertex $w \in V$
$u \leftarrow BFS(w)$
$v \leftarrow BFS(u)$
Select the vertices u_0, v_0 s.t. $distance(u_0, v_0)$ is maximal.

1. This procedure gives a vertex u_0 such that : $exc(u_0) \leq diam(G)$ i.e. a lower bound of the diameter.

1. This procedure gives a vertex u_0 such that : $exc(u_0) \leq diam(G)$ i.e. a lower bound of the diameter.
2. Use a spanning tree as a subgraph on the same vertex set to obtain an upper bound by computing its exact diameter in linear time (using the trivial bound $diam(G) \leq diam(G - e)$).

1. This procedure gives a vertex u_0 such that :
   $exc(u_0) \leq diam(G)$ i.e. a lower bound of the diameter.
2. Use a spanning tree as a subgraph on the same vertex set to obtain an upper bound by computing its exact diameter in linear time (using the trivial bound $diam(G) \leq diam(G - e)$).
3. Spanning trees given by the BFS.

- The Program and some Data on Web graphs or P-2-P networks can be found

- The Program and some Data on Web graphs or P-2-P networks can be found
- `http://www-rp.lip6.fr/~magnien/Diameter`

- The Program and some Data on Web graphs or P-2-P networks can be found
- `http://www-rp.lip6.fr/~magnien/Diameter`
- 2 millions of vertices, diameter 32 within 1

- The Program and some Data on Web graphs or P-2-P networks can be found
- `http://www-rp.lip6.fr/~magnien/Diameter`
- 2 millions of vertices, diameter 32 within 1
- Further experimentations by Crescenzi, Grossi, Marino (in ESA 2010)
  which confirm the excellence of the lower bound using BFS ! ! ! !

- Since α is a constant (≤ 1000), this method is still in linear time and works extremely well on huge graphs (Web graphs, Internet ...)

- Since α is a constant (≤ 1000), this method is still in linear time and works extremely well on huge graphs (Web graphs, Internet ...)
- How can we explain the success of such a method ?

- Since α is a constant (≤ 1000), this method is still in linear time and works extremely well on huge graphs (Web graphs, Internet . . .)
- How can we explain the success of such a method ?
- Due to the many counterexamples for the 2 consecutive BFS procedure. An explanation is necessary !

2 kind of explanations

The method is good or the data used was good.

2 kind of explanations

The method is good or the data used was good.

Partial answer

The method also works on several models of random graphs.
So let us try to prove the first fact

2 kind of explanations

The method is good or the data used was good.

Partial answer

The method also works on several models of random graphs.
So let us try to prove the first fact

Restriction

First we are going to focus our study on the 2 consecutive BFS.

Chordal graphs

1. A graph is chordal if it has no chordless cycle of length ≥ 4 .

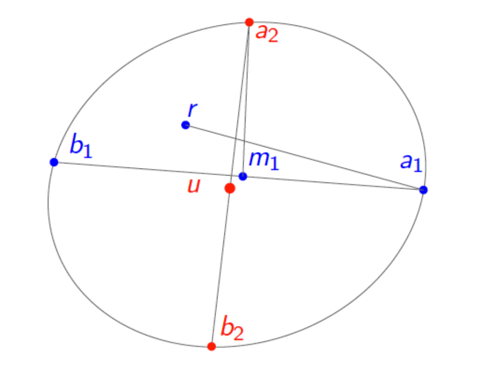
Chordal graphs

1. A graph is chordal if it has no chordless cycle of length ≥ 4 .
2. If G is a chordal graph, Corneil, Dragan, MH, Paul 2001, using a variant called 2 consecutive LexBFS
   $d(u, v) \leq diam(G) \leq d(u, v) + 1$

Chordal graphs

1. A graph is chordal if it has no chordless cycle of length ≥ 4 .
2. If G is a chordal graph, Corneil, Dragan, MH, Paul 2001, using a variant called 2 consecutive LexBFS
   $d(u, v) \leq diam(G) \leq d(u, v) + 1$
3. Generalized by Corneil, Dragan, Kohler 2003 using 2 consecutive BFS :
   $d(u, v) \leq diam(G) \leq d(u, v) + 1$

The 4-sweep : Crescenzi, Grossi, MH, Lanzi, Marino 2011



$$Diam = max\{ecc(a_1), ecc(a_2)\} \text{ and } Rad = min\{ecc(r), ecc(m_1)\}$$

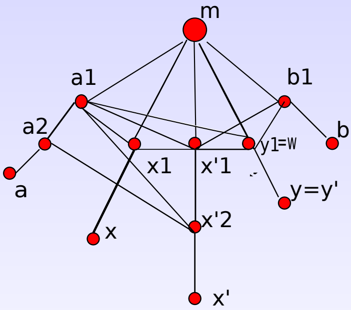
Intuition behind the 4-sweep heuristics

- Chepoi and Dragan has proved that for chordal graphs that a center is at distance at most one of the middle vertex (m_1 in the picture).

Intuition behind the 4-sweep heuristics

- Chepoi and Dragan has proved that for chordal graphs that a center is at distance at most one of the middle vertex (m_1 in the picture).
- Roughly, we have the same results with 4-sweep than with 1000 2-sweep.

It is still not al algorithm, even for chordal graph ! !



w=y1, ecc(w)=d[w,x]=3

ecc(x)=d[x,y]=4 , m is midpoint of [x,y].

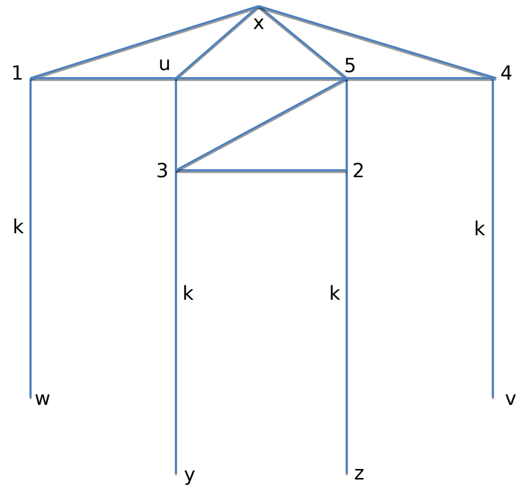
ecc(m) =d[m,x']= 3,

ecc(x')=4=d[x',b]=d[x',x]=d[x',y]

Output of 4-BFS : 4

But , Diameter=d[a,b]=5

it could be hard to be sure



An exact algorithm !

Compute the excentricity in a bottom up fashion starting from the leaves of a BFS rooted in m_1
with a stoping condition.
Complexity is $O(nm)$ in the worst case, but often linear in practice.

Simple Lemma

If for some $x \in Level(i)$ of the tree, we have $ecc(x) > 2(i-1)$ then we can stop the exploration.

Simple Lemma

If for some $x \in Level(i)$ of the tree, we have $ecc(x) > 2(i-1)$ then we can stop the exploration.

Proof

Let us consider $y \in L(j)$ with $j < i$. $\forall z \in \cup_{1 \leq k \leq i-1} L(k)$
$dist(z, y) \leq 2(i-1)$
Therefore $ecc(y) \leq ecc(x)$ or the extreme vertices from y belong to lower layers and have already been considered.

iFub an exact $O(mn)$ algorithm

Algorithm 1: iFUB (iterative Fringe Upper Bound)

Input: G, u, lower bound l

Output: A value M such that $D - M \leq k$.

$i \leftarrow \mathrm{ecc}(u)$; $lb \leftarrow \max\{\mathrm{ecc}(u), l\}$; $ub \leftarrow 2\mathrm{ecc}(u)$;

while $ub \neq lb$ **do**

  if $\max\{B_1(u), \ldots, B_i(u)\} > 2(i-1)$ **then**

    return $\max\{B_1(u), \ldots, B_i(u)\}$;

  else

    $lb \leftarrow \max\{B_1(u), \ldots, B_i(u)\}$;

    $ub \leftarrow 2(i-1)$;

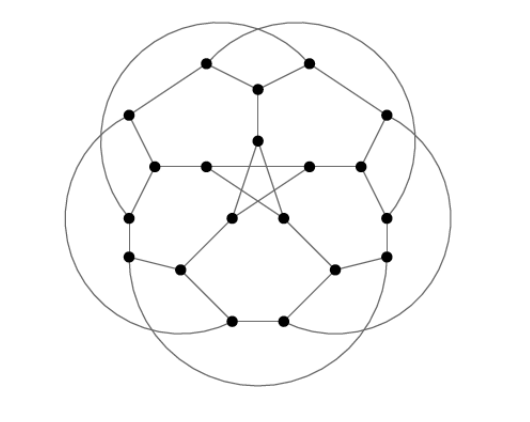
  end

  $i \leftarrow i - 1$;

end

return lb;

Bad example



Comments

- Diameter of Facebook graph = 41, Average distance 4.74, Backstrom, Boldi, Rosa, Uganden, Vigna 2011 Boldi and his group had to parallelize our algorithm and a BFS on the giant connected component of Facebook would take several hours. But only 17 BFS's were needed, so just extra 13 sweeps to certify the 4-sweep value.

Comments

- Diameter of Facebook graph = 41, Average distance 4.74, Backstrom, Boldi, Rosa, Uganden, Vigna 2011 Boldi and his group had to parallelize our algorithm and a BFS on the giant connected component of Facebook would take several hours. But only 17 BFS's were needed, so just extra 13 sweeps to certify the 4-sweep value.
- The 4-sweep method "always" gives a lower bound of the diameter not too far from the optimal,
  the hard part is to obtain an upper bound with iFUB

Comments

- Diameter of Facebook graph = 41, Average distance 4.74, Backstrom, Boldi, Rosa, Uganden, Vigna 2011 Boldi and his group had to parallelize our algorithm and a BFS on the giant connected component of Facebook would take several hours. But only 17 BFS's were needed, so just extra 13 sweeps to certify the 4-sweep value.
- The 4-sweep method "always" gives a lower bound of the diameter not too far from the optimal,
  the hard part is to obtain an upper bound with iFUB
- The worst examples are roadmap graphs with big treewidth and big grids.

Stanford Large Network Dataset Collection
http ://snap.stanford.edu/data/

- A very practical database for having large graphs to play with.

Stanford Large Network Dataset Collection
http ://snap.stanford.edu/data/

- A very practical database for having large graphs to play with.
- Graphs are described that way : number of vertices, number of edges (arcs), diameter.

Graph	diam SNAP	diam 4-Sweep
soc-Epinions1	14	15
soc-pokec-relationships	11	14
soc-Slashdot0811	10	12
soc-Slashdot0902	11	13
com-lj.ungraph	17	21
com-youtube.ungraph	20	24
com-DBLP	21	23
com-amazon	44	47
email-Enron	11	13
wikiTalk	9	11
cit-HepPh	12	14
cit-HepTh	13	15
CA-CondMat	14	15
CA-HepTh	17	18
web-Google	21	24

Graph	diam SNAP	diam 4-Sweep
amazon0302	32	38
amazon0312	18	20
amazon0505	20	22
amazon0601	21	25
p2p-Gnutella04	9	10
p2p-Gnutella24	10	11
p2p-Gnutella25	10	11
p2p-Gnutella30	10	11
roadNet-CA	849	865
roadNet-TX	1054	1064
Gowalla-edges	14	16
BrightKite-edges	16	18

How can I certify my results ?

- How can I beat the value of Stanford database ?

How can I certify my results ?

- How can I beat the value of Stanford database ?
- Then some * explains in a little footnote that the SNAP value is heuristically obtained by 1000 random BFS

How can I certify my results ?

- How can I beat the value of Stanford database ?
- Then some * explains in a little footnote that the SNAP value is heuristically obtained by 1000 random BFS
- I like the idea that 4 searches totally dependent are better that 1000 independent searches

How can I certify my results ?

- How can I beat the value of Stanford database ?
- Then some * explains in a little footnote that the SNAP value is heuristically obtained by 1000 random BFS
- I like the idea that 4 searches totally dependent are better that 1000 independent searches
- See the example of a long path.

How can I certify my results ?

- How can I beat the value of Stanford database ?
- Then some * explains in a little footnote that the SNAP value is heuristically obtained by 1000 random BFS
- I like the idea that 4 searches totally dependent are better that 1000 independent searches
- See the example of a long path.
- The last vertex of a BFS is not at all a random vertex (NP-complete to decide : Charbit, MH, Mamcarz 2014 DMTCS).

How can I certify my results ?

- By certifying the longest path $[x, y]$ (as hard as computing a BFS ?)

How can I certify my results ?

- By certifying the longest path $[x, y]$ (as hard as computing a BFS ?)
- Using another BFS programmed by others starting at x.

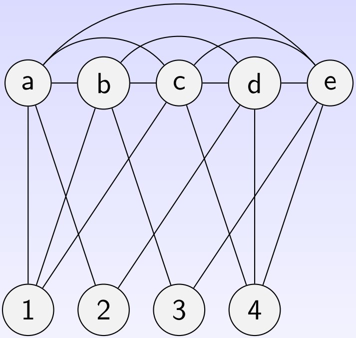
How can I certify my results ?

- By certifying the longest path $[x, y]$ (as hard as computing a BFS ?)
- Using another BFS programmed by others starting at x.
- Certifying that the computed BFS ordering is a legal BFS ordering, using the 4-points condition. Which can be checked in linear time for BFS and DFS.

Graph Name	$\frac{\mathit{Vertices}}{\mathit{Edges}}$	Diameter iFUB	Diam. FourSweep
CA-HepTh	0.190	18	18
CA-GrQc	0.181	17	17
CA-CondMat	0.124	15	15
CA-AstroPh	0.047	14	14
roadNet-CA	0.355	865	865
roadNet-PA	0.353	794	780
roadNet-TX	0.359	1064	1064
email-Enron	0.1	13	13
email-EuAll	0.631	14	14
com-amazon	0.361	47	47
Amazon0302	0.212	38	38
Amazon0312	0.125	20	20
Amazon0505	0.122	22	22
Amazon0601	0.119	25	25
Gowalla_edges	0.207	25	16
Brightkite_edges	0.272	18	18
soc-Epinions1	0.149	15	15

FIGURE: 4-Sweep Results

Multisweep of BFS or even LexBFS are not enough



$$\text{LexBFS} = 4\ e\ d\ c\ b\ a\ 3\ 2\ 1 = \tau$$
$$\text{LexBFS}^{+}(G, \tau) = 1\ a\ b\ c\ d\ e\ 2\ 3\ 4 = \theta$$
$$\text{LexBFS}^{+}(G, \theta) = \tau\ \ldots$$

The example

So in this split graph LexBFS$^+$ will infinitely loop between τ and θ proposing as extremal vertices 1 and 4 with d(1,4)=2
but the diameter is three with d(2,3)=3. Even a probabilistic argument does not hold since we could add many twins of vertices 1 and 4.

The example

So in this split graph LexBFS$^+$ will infinitely loop between τ and θ proposing as extremal vertices 1 and 4 with d(1,4)=2
but the diameter is three with d(2,3)=3. Even a probabilistic argument does not hold since we could add many twins of vertices 1 and 4.

By the way

We (Pierre, Reza, Lalla and I) have a conjecture that for all split graphs, LexBFS$^+$ always loops between two permutations.

So we have two main ingredients

So we have two main ingredients

- BFS and triangle inequality

So we have two main ingredients

- BFS and triangle inequality
- How to use them cleverly ?

Methods which maintain an interval $[ecc_-(x), ecc_+(x)]$ for each vertex

Methods which maintain an interval $[ecc_{-}(x), ecc_{+}(x)]$ for each vertex

- Starting with $[0, \infty]$ for every vertex

Methods which maintain an interval $[ecc_{-}(x), ecc_{+}(x)]$ for each vertex

- Starting with $[0, \infty]$ for every vertex
- After computing a BFS starting at a vertex u we have computed not only $ecc(u)$ but also $\forall x, d(u, x)$.

Methods which maintain an interval $[ecc_-(x), ecc_+(x)]$ for each vertex

- Starting with $[0, \infty]$ for every vertex
- After computing a BFS starting at a vertex u we have computed not only $ecc(u)$ but also $\forall x, d(u, x)$.
- Since $d(x, u) \leq ecc(x) \leq ecc(u) + d(u, x)$

Methods which maintain an interval $[ecc_-(x), ecc_+(x)]$ for each vertex

- Starting with $[0, \infty]$ for every vertex
- After computing a BFS starting at a vertex u we have computed not only $ecc(u)$ but also $\forall x, d(u, x)$.
- Since $d(x, u) \leq ecc(x) \leq ecc(u) + d(u, x)$
- We can update $ecc_-(x)$ with $d(x, u)$ and $ecc_+(x)$ with $ecc(u) + d(u, x)$.

Halting conditions

Halting conditions

For all eccentricities Until $\forall x, ecc_-(x) = ecc_+(x)$

Halting conditions

For all eccentricities Until $\forall x, ecc_{-}(x) = ecc_{+}(x)$

For diameter After a BFS starting at v such that : $\forall x, ecc_{+}(x) \leq ecc(v)$

Halting conditions

For all eccentricities Until $\forall x, ecc_-(x) = ecc_+(x)$

For diameter After a BFS starting at v such that : $\forall x, ecc_+(x) \leq ecc(v)$

For radius After a BFS starting at v such that : $\forall x, ecc(v) \leq ecc_-(x)$

They are several algorithms within this framework.
The only thing we did not fix yet is the choice of the next vertex to start a BFS.

Sumsweep : a symmetric method for computing radius and diameter

M. Borassi, P. Crescenzi, R. Grossi, MH, W. Kosters, A. Marino and F. Takes, 2014

- A mixture with our approach and that of W. Kosters and F. Takes in which a lower bound of the eccentricity of every vertex is maintained at each BFS.

Sumsweep : a symmetric method for computing radius and diameter

M. Borassi, P. Crescenzi, R. Grossi, MH, W. Kosters, A. Marino and F. Takes, 2014

- A mixture with our approach and that of W. Kosters and F. Takes in which a lower bound of the eccentricity of every vertex is maintained at each BFS.
- It generalizes the 4-sweep method to k-sweep.

Sumsweep : a symmetric method for computing radius and diameter

M. Borassi, P. Crescenzi, R. Grossi, MH, W. Kosters, A. Marino and F. Takes, 2014

- A mixture with our approach and that of W. Kosters and F. Takes in which a lower bound of the eccentricity of every vertex is maintained at each BFS.
- It generalizes the 4-sweep method to k-sweep.
- we generalize to maintain k values in each vertex.

- Given a random vertex v_1 and setting $i = 1$, repeat k times the following :
  1. Perform a BFS from v_i and choose the vertex v_{i+1} as the vertex x maximizing $\sum_{j=1}^{i} d(v_j, x)$.
  2. Perform a BFS starting at a vertex v_{i+1}, as the vertex w minimizing $\sum_{i=1}^{k} d(w, v_i)$.
  3. Increment i.
- The maximum eccentricity found, i.e. $\max_{i=1,\dots,k} exc(v_i)$, is a lower bound for the diameter.
- The minimum eccentricity found, i.e. $\min\{\min_{i=1,\dots,k} exc(v_i), exc(w)\}$, is an upper bound for the radius.

Halting conditions

To compute the exact values of radius and diameter, we use the next lemmas.

Lemma 1

Let $Diam(G)$ be the diameter, let x and y be diametral vertices (that is, $d(x, y) = Diam(G)$), and let $v_1, \ldots, v_k$ be k other vertices. Then, $Diam(G) \leq \frac{2}{k} \sum_{i=1}^{k} d(x, v_i)$ or $Diam(G) \leq \frac{2}{k} \sum_{i=1}^{k} d(y, v_i)$.

Halting conditions

To compute the exact values of radius and diameter, we use the next lemmas.

Lemma 1

Let $Diam(G)$ be the diameter, let x and y be diametral vertices (that is, $d(x,y) = Diam(G)$), and let $v_1, \ldots, v_k$ be k other vertices. Then, $Diam(G) \leq \frac{2}{k}\sum_{i=1}^{k} d(x, v_i)$ or $Diam(G) \leq \frac{2}{k}\sum_{i=1}^{k} d(y, v_i)$.

proof

$kDiam(G) = \sum_{i=1}^{k} d(x,y) \leq \sum_{i=1}^{k} [d(x,v_i) + d(v_i,y)] =$
$\sum_{i=1}^{k} d(x,v_i) + \sum_{i=1}^{k} d(y,v_i)$.
If $\sum_{i=1}^{k} d(x,v_i) \leq \sum_{i=1}^{k} d(y,v_i)$ then $Diam(G) \leq \frac{2}{k}\sum_{i=1}^{k} d(y,v_i)$.
If $\sum_{i=1}^{k} d(y,v_i) \leq \sum_{i=1}^{k} d(x,v_i,)$ then
$Diam(G) \leq \frac{2}{k}\sum_{i=1}^{k} d(x,v_i)$. $\square$

corollary 1

During the algorithm, if for a vertex z there exists a vertex t such that $ecc(t) > 2/k \sum_{i=1}^{k} d(z, v_i)$ then $ecc(z) < Diam(G)$.

corollary 1

During the algorithm, if for a vertex z there exists a vertex t such that $ecc(t) > 2/k \sum_{i=1}^{k} d(z, v_i)$ then $ecc(z) < Diam(G)$.

proof

We have $Diam(G) \geq ecc(t) > 2/k \sum_{i=1}^{k} d(z, v_i)$
But if z was an extremal vertex we would have using the previous lemma :
$2/k \sum_{i=1}^{k} d(z, v_i) \geq Diam(G)$, a contradiction.

$\square$

Lemma 2

Let $x \in V$ be a center and let $v_1, \ldots, v_k$ be k other vertices. Then $Radius(G) \geq 1/k \sum_{i=1}^{k} d(x, v_i)$

Lemma 2

Let $x \in V$ be a center and let $v_1, \ldots, v_k$ be k other vertices. Then $Radius(G) \geq 1/k \sum_{i=1}^{k} d(x, v_i)$

proof

Let $y \in V$ such that : $Radius(G) = d(x, y)$
Then $kRadius(G) = \sum_{i=1}^{k} d(x, y) \geq \sum_{i=1}^{k} d(x, v_i)$. Since $d(x, y)$ is the exccentricity of x, we have : $\forall v_i \; d(x, y) \geq d(x, v_i)$.

$\square$

corollary 2

During the algorithm, if for a vertex z there exists a vertex t such that $ecc(t) < 1/k \sum_{i=1}^{k} d(z, v_i)$ then z cannot be a center of G.

corollary 2

During the algorithm, if for a vertex z there exists a vertex t such that $ecc(t) < 1/k \sum_{i=1}^{k} d(z, v_i)$ then z cannot be a center of G.

proof

We have
$Radius(G) \leq ecc(t) < d(x, y) < 1/k \sum_{i=1}^{k} d(z, v_i) \leq ecc(z)$
Therefore $Radius(G) < ecc(z)$. □

- If during the algorithm we maintain two variables Macsofar and Minsofar (being respectively the maximum and the minimum computed eccentricity)

- If during the algorithm we maintain two variables Macsofar and Minsofar (being respectively the maximum and the minimum computed eccentricity)
- To compute a diameter we can only perform a BFS starting at x such that $Maxsofar \leq \frac{2}{k}\sum_{i=1}^{k} d(x, v_i)$ (using corollary 1).

- If during the algorithm we maintain two variables Macsofar and Minsofar (being respectively the maximum and the minimum computed eccentricity)
- To compute a diameter we can only perform a BFS starting at x such that $Maxsofar \leq \frac{2}{k} \sum_{i=1}^{k} d(x, v_i)$ (using corollary 1).
- To compute a radius we can only perform a BFS starting at x such that : $1/k \sum_{i=1}^{k} d(x, y) \leq Minsofar$ (using corollary 2).

- This method generalizes the 4-sweep and seems to better handle the cases where 1000 BFS was needed to find the exact value in the previous method.

- This method generalizes the 4-sweep and seems to better handle the cases where 1000 BFS was needed to find the exact value in the previous method.
- For the same examples it never goes further 10-100 BFS.

- This method generalizes the 4-sweep and seems to better handle the cases where 1000 BFS was needed to find the exact value in the previous method.
- For the same examples it never goes further 10-100 BFS.
- Strangely replacing Sum by Max as suggested by some experts does not change the behavior of the algorithm.

Easy extensions

1. Since we only use a way to compute all distances form a given source and triangle inequality.

Easy extensions

1. Since we only use a way to compute all distances form a given source and triangle inequality.
2. To weighted graphs by replacing BFS with Dijkstra's algorithm

Easy extensions

1. Since we only use a way to compute all distances form a given source and triangle inequality.
2. To weighted graphs by replacing BFS with Dijkstra's algorithm
3. To directed graphs with directed eccentricity.

(Sumsweep) M. Borassi, P. Crescenzi, R. Grossi, M. Habib, W. Kosters, A. Marino and F. Takes, *Fast diameter and radius BFS-based computation in (weakly connected) real-world graphs : With an application to the six degrees of separation games*, **Theor. Comput. Sci.** 586 : 59-80 (2015).

Real Applications with Sumsweep

With this method we were able to disprove conjectures inspired from S. Milgram about the 6 degrees of separation

1. Kevin Bacon games on the actors graph

Real Applications with Sumsweep

With this method we were able to disprove conjectures inspired from S. Milgram about the 6 degrees of separation

1. Kevin Bacon games on the actors graph
2. Diameter of Wikipedia (the Wiki Game)

Kevin Bacon

His name was used for a popular TV game in US, The Six Degrees of Kevin Bacon, in which the goal is to connect an actor to Kevin Bacon in less than 6 edges.

Actors graph 2014

- The 2014 graph has 1.797.446 in the biggest connected component, a few more if we consider the whole graph. The number of undirected edges in the biggest connected component is 72.880.156.

Actors graph 2014

- The 2014 graph has 1.797.446 in the biggest connected component, a few more if we consider the whole graph. The number of undirected edges in the biggest connected component is 72.880.156.
- An actor with Bacon number 8 is Shemise Evans, and the path can be found at http ://oracleofbacon.org/ by writing Shemise Evans in the box. Even if their graph does not coincide exactly with our graph, this is a shortest path in both of them :

Shemise Evans → Casual Friday (2008) → Deniz Buga
Deniz Buga → Walking While Sleeping (2009)→ Onur Karaoglu
Onur Karaoglu→ Kardesler (2004)→ Fatih Genckal
Fatih Genckal → Hasat (2012) → Mehmet Ünal
Mehmet Ünal→ Kayip özgürlük (2011)→ Aydin Orak
Aydin Orak → The Blue Man (2014)→Alex Dawe
Alex Dawe→ Taken 2 (2012)→ Rade Serbedzija
Rade Serbedzija→ X-Men : First Class (2011) → Kevin Bacon

Twitter graph (2011)

Directed graph with 500 million vertices and 2,5 billion edges.
We found the exact value of the diameter (150) of the giant connected component (computed in 2016).

Radius versus diameter

- Let D, R be respectively two potential values for $diam(G)$ and $radius(G)$.

Radius versus diameter

- Let D, R be respectively two potential values for $diam(G)$ and $radius(G)$.
- To certify these values we need to prove :

Radius versus diameter

- Let D, R be respectively two potential values for $diam(G)$ and $radius(G)$.
- To certify these values we need to prove :
- $\forall x \in V(G)$, $\forall y \in V(G)$, we have $d(x, y) \leq D$.

Radius versus diameter

- Let D, R be respectively two potential values for $diam(G)$ and $radius(G)$.
- To certify these values we need to prove :
- $\forall x \in V(G)$, $\forall y \in V(G)$, we have $d(x,y) \leq D$.
- $\forall x \in V(G)$, $\exists y \in V(G)$ such that $d(x,y) \geq R$.

Radius versus diameter

- Let D, R be respectively two potential values for $diam(G)$ and $radius(G)$.
- To certify these values we need to prove :
- $\forall x \in V(G)$, $\forall y \in V(G)$, we have $d(x,y) \leq D$.
- $\forall x \in V(G)$, $\exists y \in V(G)$ such that $d(x,y) \geq R$.
- Not exactly the same quantifiers !

Relationships between diameter and δ-hyperbolicity

δ-Hyperbolic metric spaces have been defined by M. Gromov in 1987 via a simple 4-point condition :
for any four points u, v, w, x, the two larger of the distance sums $d(u,v)+d(w,x), d(u,w)+d(v,x), d(u,x)+d(v,w)$ differ by at most 2δ.
$\delta - hyberbolicity(G) = 0$ iff G is a tree.

Nice

Because many real networks have small δ-hyperbolicity.
In fact δ-hyperbolicity captures the distance of a graph to a tree in a metric way.

Theorem Chepoi, Dragan, Estellon, M.H., Vaxes 2008

If u is the last vertex of a 2-sweep then :
$exc(u) \geq diam(G)$-2.$\delta(G)$ and
$radius(G) \leq \lceil (d(u, v) + 1)/2 \rceil + 3\delta(G)$
Furthermore the set of all centers $C(G)$ of G is contained in the ball of radius $5\delta(G) + 1$ centered at a middle vertex m of any shortest path connecting u and v in G.

Theorem Chepoi, Dragan, Estellon, M.H., Vaxes 2008

If u is the last vertex of a 2-sweep then :
$exc(u) \geq diam(G)$-$2.\delta(G)$ and
$radius(G) \leq \lceil (d(u,v)+1)/2 \rceil + 3\delta(G)$
Furthermore the set of all centers $C(G)$ of G is contained in the ball of radius $5\delta(G)+1$ centered at a middle vertex m of any shortest path connecting u and v in G.

Consequences

The 2-sweep (resp 4-sweep) method error is bounded by the δ-hyperbolicity of the graph.

Definition

A graph G is a split graph, if its vertex set $V(G)$ can be partitioned into a maximal clique K and a stable set S.

- The diameter of a non-complete split graph can only be either 2 or 3.

- The diameter of a non-complete split graph can only be either 2 or 3.
- In fact computing diameter, i.e. deciding between 2 or 3, is already hard for split graphs !

- The diameter of a non-complete split graph can only be either 2 or 3.
- In fact computing diameter, i.e. deciding between 2 or 3, is already hard for split graphs !
- And for years I have tried to find a linear time algorithm to compute the exact diameter of split graphs ....

- The diameter of a non-complete split graph can only be either 2 or 3.
- In fact computing diameter, i.e. deciding between 2 or 3, is already hard for split graphs !
- And for years I have tried to find a linear time algorithm to compute the exact diameter of split graphs ....
- We immediately noticed the equivalence to the disjoint set problem

Disjoint subsets problem

Disjoint sets problem

A finite set X, $\mathcal{F}$ a collection $\{S_1, \ldots, S_k\}$ of subsets of X.
$\exists i, j \in [1, k]$ such that $S_i \cap S_j = \emptyset$?

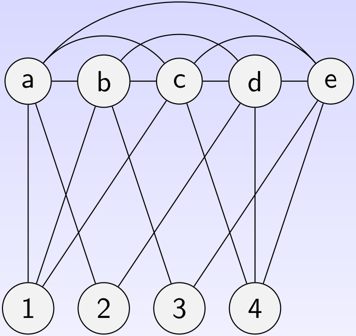
Disjoint subsets problem

Disjoint sets problem

A finite set X, $\mathcal{F}$ a collection $\{S_1, \dots, S_k\}$ of subsets of X.
$\exists i, j \in [1, k]$ such that $S_i \cap S_j = \emptyset$?

Linearity

Can this problem be solved in linear time ?
Size of the problem : $|X| + k + \sum_{i=1}^{i=k} |S_i|$
size of the incidence bipartite graph



$\text{LexBFS} = 4\ e\ d\ c\ b\ a\ 3\ 2\ 1 = \tau$

$\text{LexBFS}^+(G, \tau) = 1\ a\ b\ c\ d\ e\ 2\ 3\ 4 = \theta$

$\text{LexBFS}^+(G, \theta) = \tau\ \ldots$

The example

So in this example LexBFS$^+$ will infinitely loop between τ and θ proposing as extremal vertices 1 and 4 with d(1,4)=2
but the diameter is three with d(2,3)=3.

The example

So in this example LexBFS$^+$ will infinitely loop between τ and θ proposing as extremal vertices 1 and 4 with d(1,4)=2
but the diameter is three with d(2,3)=3.

By the way

We (Pierre, Reza, Lalla and I) have a conjecture that for all split graphs, LexBFS$^+$ always loops between two permutations.

SETH : Strong Exponential Time Hypothesis

SETH

SETH : Strong Exponential Time Hypothesis

SETH

- There is no algorithm for solving the k-SAT problem with n variables in $O((2-\epsilon)^n)$ where ϵ does not depend on k.

SETH : Strong Exponential Time Hypothesis

SETH

- There is no algorithm for solving the k-SAT problem with n variables in $O((2-\epsilon)^n)$ where ϵ does not depend on k.
- Since it is not true for $k = 3$, the conjecture must be precised as follows :

SETH : Strong Exponential Time Hypothesis

SETH

- There is no algorithm for solving the k-SAT problem with n variables in $O((2-\epsilon)^n)$ where ϵ does not depend on k.
- Since it is not true for $k=3$, the conjecture must be precised as follows :
- For every $\epsilon > 0$, there exists an integer $q > 0$ such that there is no algorithm in $O((2-\epsilon)^n)$ for q-SAT.

Related work (1/3) : Lower-bounds

Under SETH (resp., OV) we cannot solve DIAMETER in truly subquadratic time (Roditty and V. Williams, *STOC'13*).

$\Longrightarrow$ hardness results obtained for *constant* diameter (*e.g.*, 2 vs. 3)

$\Longrightarrow$ also holds for split graphs (Borassi et al., *ETCS'16*)

Let us consider an instance I of $k - SAT$ with 2n boolean variables $x_1, \ldots, x_{2n}$, and a set $\mathcal{C}$ of m clauses $C_1, \ldots C_m$, we build an instance of Disjoint-set problem as follows :

- The set C is the set of clauses (a vertex per clause) + 2 extras vertices a, b.

Let us consider an instance I of $k-SAT$ with 2n boolean variables $x_1, \ldots, x_{2n}$, and a set $\mathcal{C}$ of m clauses $C_1, \ldots C_m$, we build an instance of Disjoint-set problem as follows :

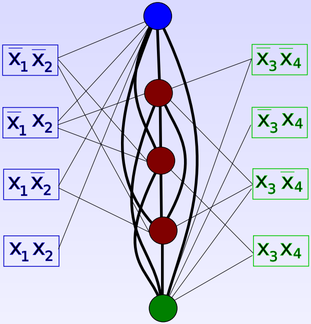
- The set C is the set of clauses (a vertex per clause) + 2 extras vertices a, b.
- We consider now A, B the sets of all truth assignments of $x_1, \ldots, x_n$, and $x_{n+1}, \ldots x_{2n}$, respectively.

Let us consider an instance I of $k - SAT$ with 2n boolean variables $x_1, \ldots, x_{2n}$, and a set $\mathcal{C}$ of m clauses $C_1, \ldots C_m$, we build an instance of Disjoint-set problem as follows :

- The set C is the set of clauses (a vertex per clause) $+$ 2 extras vertices a, b.
- We consider now A, B the sets of all truth assignments of $x_1, \ldots, x_n$, and $x_{n+1}, \ldots x_{2n}$, respectively.
- For each truth t assigment in A (resp. in B) we define $S_t = \{C \in \mathcal{C}$ such that t does not satisfy $C\} \cup \{a\}$ (resp. $\cup\{b\}$).
  S_t are the neighbours of t in $C \cup \{a, b\}$.

Let us consider an instance I of $k - SAT$ with 2n boolean variables $x_1, \ldots, x_{2n}$, and a set $\mathcal{C}$ of m clauses $C_1, \ldots C_m$, we build an instance of Disjoint-set problem as follows :

- The set C is the set of clauses (a vertex per clause) $+ 2$ extras vertices a, b.
- We consider now A, B the sets of all truth assignments of $x_1, \ldots, x_n$, and $x_{n+1}, \ldots x_{2n}$, respectively.
- For each truth t assigment in A (resp. in B) we define $S_t = \{C \in \mathcal{C}$ such that t does not satisfy $C\} \cup \{a\}$ (resp. $\cup\{b\}$).
  S_t are the neighbours of t in $C \cup \{a, b\}$.
- We add a clique on $X \cup \{a, b\}$. The vertices corresponding to the partial truth assigments correspond the independent set.



$$(x_1 \vee \bar{x}_2 \vee x_4) \wedge (x_1 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (x_2 \vee \bar{x}_3 \vee x_4)$$

- The sets $S's$ defined with A (resp. B) always intersect because of a (resp. b).

- The sets $S's$ defined with A (resp. B) always intersect because of a (resp. b).
- If for every $u \in A$ and $v \in B$, $S_u \cap S_v \supseteq \{C_j\}$, it means that for every (u, v) there exists a clause C_j which is not satisfiable.

- The sets $S's$ defined with A (resp. B) always intersect because of a (resp. b).
- If for every $u \in A$ and $v \in B$, $S_u \cap S_v \supseteq \{C_j\}$, it means that for every (u, v) there exists a clause C_j which is not satisfiable.
- If there exists S_u, S_v that do not intersect. Necessarily u is a truth assignment in A and v in B (or the converse, but they cannot be on the same set of variables because of the dummy vertices a, b).

  This means that for each clause C_i of I, if $C_i \notin S_u$, then the truth u assignment satisfies C_i.

  Similarly if $C_i \notin S_v$, then the truth v assignment satisfies C_i.

  But $S_u \cap S_v = \emptyset$ means that for every clause C_i either :

  $C_i \notin S_u$ or $C_i \notin S_v$.

Therefore :

I is satisfiable iff there exist 2 disjoint sets S_u, S_v.

Complexity issues

- Size of the $k - SAT$ instance is bounded by :
  $K = 2n + m + km$

Complexity issues

- Size of the $k - SAT$ instance is bounded by :
  $K = 2n + m + km$
- Size of the Disjoint set instance :
  $N = 2^{n+1} + m + 2$ vertices
  and at most $M = m2^{n+1}$ edges.

Complexity issues

- Size of the $k - SAT$ instance is bounded by :
  $K = 2n + m + km$
- Size of the Disjoint set instance :
  $N = 2^{n+1} + m + 2$ vertices
  and at most $M = m2^{n+1}$ edges.
- To compute this instance we need to evaluate the m, k-clauses for each half-truth assigment.
  Can be done in $O(K)$, so in the whole : $O(2^{n+1}K)$.

Complexity issues

- Size of the $k-SAT$ instance is bounded by :
  $K = 2n + m + km$
- Size of the Disjoint set instance :
  $N = 2^{n+1} + m + 2$ vertices
  and at most $M = m2^{n+1}$ edges.
- To compute this instance we need to evaluate the m, k-clauses for each half-truth assigment.
  Can be done in $O(K)$, so in the whole : $O(2^{n+1}K)$.
- If there exists an algorithm for the Disjoint set problem in less than $O(NM^{1-\epsilon})$
  it would imply an algorithm for $k-SAT$ in less than $O((2-\epsilon)^{2n})$ contradicting the SETH.

Consequences

Practically it means that there is less hope to design a linear time algorithm for :

- Disjoint set problem

Consequences

Practically it means that there is less hope to design a linear time algorithm for :

- Disjoint set problem
- Diameter computations for chordal graphs and split graphs

Consequences

Practically it means that there is less hope to design a linear time algorithm for :

- Disjoint set problem
- Diameter computations for chordal graphs and split graphs
- And many other related problems . . . such as betweenness centrality, see Borassi et al 2017.

Consequences

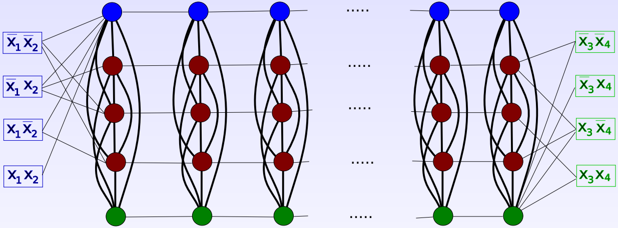
Practically it means that there is less hope to design a linear time algorithm for :

- Disjoint set problem
- Diameter computations for chordal graphs and split graphs
- And many other related problems . . . such as betweenness centrality, see Borassi et al 2017.
- but not all $O(mn)$ problems as for example transitive closure, existence of a triangle . . .

(Using Sparsification Lemma) $\forall \varepsilon, \exists c$ s.t. we cannot solve DIAMETER in $\mathcal{O}(n^{2-\varepsilon})$-time on split graphs with clique-number $\leq c \cdot \log n$.

One can play with the previous construction

Repeat n times the clique of an arbitrary split graph
(matching between every two consecutive copies)



$\forall\varepsilon, \exists c$ s.t. we cannot solve $(\frac{1}{c\cdot\log n})$-Diameter in $\mathcal{O}(n^{2-\varepsilon})$-time.

Also there are several variations on the SETH (as for example ETH).

A huge domain called Hardness in P proving lower bounds for polynomial problems using algorithmic conjectures.

As we have seen, the main tool is reductions between problems.

There are many interesting subclasses of split graphs

- threshold graphs, comparability split graphs, interval split graphs, cocomparability split graphs . . .

There are many interesting subclasses of split graphs

- threshold graphs, comparability split graphs, interval split graphs, cocomparability split graphs . . .
- Can we found some boarder between linear and non-linear ?

We start by analyzing the following very simple algorithm on split graphs :

Require: A split graph G.

```
if G has maximum degree |V(G)| - 1 then
    /* universal vertex */
    output "diam(G) ≤ 2"
else
    output "diam(G) = 3".
end if
```

Lemma

For every $G = (V, E)$ and $u, v \in V$ such that u is a maximum neighbour of v, we have $diam(G) \leq 2$ if and only if u is a universal vertex.

Lemma

For every $G = (V, E)$ and $u, v \in V$ such that u is a maximum neighbour of v, we have $diam(G) \leq 2$ if and only if u is a universal vertex.

Corollary

It turns out that many well-structured graph classes ensure the existence of a vertex with a maximum neighbour such as : graphs with a pendant vertex, threshold graphs, strongly chordal graphs and interval graphs, or even more generally dually chordal graphs. For all these classes, the above Algorithm 1 is a linear-time procedure for diameter computation on the corresponding split subclass. We can compute the diameter of split graphs with minimum degree one and dually chordal split graphs in linear time. In particular, we can compute the diameter of interval split graphs and strongly chordal split graphs in linear time.

We say that G is *clique-interval* if there exists a total ordering over K such that, for every $v \in S$, the vertices in $N_G(v)$ are consecutive.

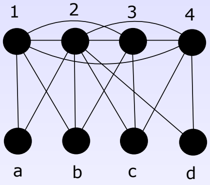


FIGURE: An interval split graph that is not clique-interval.

Proposition

A clique-interval split graph has diameter at most two if and only if it has a universal vertex. In particular, we can compute the diameter of clique-interval split graphs in linear time.

- A graph $G = (V, E)$ is called k-interval if we can map every $v \in V$ to the union of at most k closed interval on the real line, denoted $I(v)$, in such a way that $uv \in E \iff I(u) \cap I(v) \neq \emptyset$. In particular, 1-interval graphs are exactly the interval graphs. These definitions apply to any graph. A *k-interval split graph* is a split graph that is k-interval.

- A graph $G = (V, E)$ is called k-interval if we can map every $v \in V$ to the union of at most k closed interval on the real line, denoted $I(v)$, in such a way that $uv \in E \iff I(u) \cap I(v) \neq \emptyset$. In particular, 1-interval graphs are exactly the interval graphs. These definitions apply to any graph. A *k-interval split graph* is a split graph that is k-interval.
- A split graph $G = (K \cup S, E)$ is called *k-clique-interval* if there exists a total order over the maximal clique K such that, for every $v \in S$ in the stable set, $N_G(v)$ is the union of at most k intervals. In particular, the 1-clique-interval split graphs are exactly the clique-interval split graphs defined previously.

- A graph $G = (V, E)$ is called k-interval if we can map every $v \in V$ to the union of at most k closed interval on the real line, denoted $I(v)$, in such a way that $uv \in E \iff I(u) \cap I(v) \neq \emptyset$. In particular, 1-interval graphs are exactly the interval graphs. These definitions apply to any graph. A *k-interval split graph* is a split graph that is k-interval.
- A split graph $G = (K \cup S, E)$ is called *k-clique-interval* if there exists a total order over the maximal clique K such that, for every $v \in S$ in the stable set, $N_G(v)$ is the union of at most k intervals. In particular, the 1-clique-interval split graphs are exactly the clique-interval split graphs defined previously.
- We recall that already for $k = 1$, we observed in the previous section that clique-interval split graphs and interval split graphs are two overlapping subclasses.

A *comparability graph* is a graph that admits a transitive orientation.

lemma

For every comparability split graph $G = (K \cup S, E)$, we can compute in linear time a total order over K such that, for every $v \in S$, $N_G(v)$ is the union of a prefix and a suffix of this order. In particular, every comparability split graph is 2-clique-interval.

Main Result.

We study the fine-grained complexity of diameter computations on split graphs with bounded interval number. Our main finding is the following dichotomy result :

Theorem

If $G = (K \cup S, E)$ is an n-vertex m-edge k-interval split graph and a corresponding interval order is given then, we can compute the diameter of G in time $\mathcal{O}(k^2(m + 2^{\mathcal{O}(k)}n^{1+o(1)}))$. This is quasi linear-time if $k = o(\log n)$.
Conversely, under SETH we cannot compute the diameter of n-vertex split graphs with interval number $\omega(\log n)$ in subquadratic time.

Additional results

1. For complements of k-clique interval split graphs we can compute the diameter in $O(km)$

Additional results

1. For complements of k-clique interval split graphs we can compute the diameter in $O(km)$
2. Recognition of clique-interval split graphs can be done in linear time.

Open questions

1. Same dichotomy result for k-clique interval ?

Open questions

1. Same dichotomy result for k-clique interval ?
2. Recognition of k-interval *split graphs*, for $k > 2$?

Open questions

1. Same dichotomy result for k-clique interval ?
2. Recognition of k-interval *split graphs*, for $k > 2$?
3. Recognition of k-clique-interval split graphs, for $k \geq 2$?

New results Soda 2020

Under the Strong Exponential-Time Hypothesis, the diameter of general unweighted graphs cannot be computed in truly subquadratic time. Nevertheless there are several graph classes for which this can be done such as bounded-treewidth graphs, interval graphs and planar graphs, to name a few.
We propose to study unweighted graphs of constant distance VC-dimension as a broad generalization of many such classes – where the distance VC-dimension of a graph G is defined as the VC-dimension of its ball hypergraph : whose hyperedges are the balls of all possible radii and centers in G.

In particular for any fixed H, the class of H-minor free graphs has distance VC-dimension at most $|V(H)| - 1$.
In SODA 2020, we show the following.

- Our first main result is a Monte Carlo algorithm that on graphs of distance VC-dimension at most d, for any fixed k, either computes the diameter or concludes that it is larger than k in time $\tilde{\mathcal{O}}(k \cdot mn^{1-\varepsilon_d})$, where $\varepsilon_{\boldsymbol{d}} \in (0;1)$ only depends on d. We thus obtain a truly subquadratic-time parameterized algorithm for computing the diameter on such graphs.

In particular for any fixed H, the class of H-minor free graphs has distance VC-dimension at most $|V(H)| - 1$.
In SODA 2020, we show the following.

- Our first main result is a Monte Carlo algorithm that on graphs of distance VC-dimension at most d, for any fixed k, either computes the diameter or concludes that it is larger than k in time $\tilde{\mathcal{O}}(k \cdot mn^{1-\varepsilon_d})$, where $\varepsilon_d \in (0;1)$ only depends on d. We thus obtain a truly subquadratic-time parameterized algorithm for computing the diameter on such graphs.
- Then as a byproduct of our approach, we get the first truly subquadratic-time randomized algorithm for *constant* diameter computation on all the nowhere dense graph classes. The latter classes include all proper minor-closed graph classes, bounded-degree graphs and graphs of bounded expansion.

- Finally, we show how to remove the dependency on k for *any* graph class that excludes a fixed graph H as a minor. More generally, our techniques apply to any graph with constant distance VC-dimension and *polynomial expansion* (or equivalently having strongly sublinear balanced separators). As a result for all such graphs one obtains a truly subquadratic-time randomized algorithm for computing their diameter.

We note that all our results also hold for radius computation. Our approach is based on the work of Chazelle and Welzl who proved the existence of spanning paths with strongly sublinear stabbing number for every hypergraph of constant VC-dimension. We show how to compute such paths efficiently by combining known algorithms for the stabbing number problem with a clever use of ε-nets, region decomposition and other partition techniques.

Perspectives

- Improved the Soda results by avoiding the randomized aspects

Perspectives

- Improved the Soda results by avoiding the randomized aspects
- Back to practical with these new ideas

Perspectives

- Improved the Soda results by avoiding the randomized aspects
- Back to practical with these new ideas
- New ! Compute shortest paths for race sailing boats using the weather forecast (i.e., shortest paths in huge temporal graphs).

Algorithmic and experimental aspects

- (2-sweep) C. Magnien, M. Latapy, M. Habib, *Fast computation of empirically tight bounds for the diameter of massive graphs*, **Journal of Experimental Algorithmics**, 13 (2008).
- (IFub) P. Crescenzi, R. Grossi, M. Habib, L. Lanzi and A. Marino, *On Computing the Diameter of Real-World Undirected graphs*, **Theor. Comput. Sci.** 514 : 84-95 (2013).
- (Sumsweep) M. Borassi, P. Crescenzi, R. Grossi, M. Habib, W. Kosters, A. Marino and F. Takes, *Fast diameter and radius BFS-based computation in (weakly connected) real-world graphs : With an application to the six degrees of separation games*, **Theor. Comput. Sci.** 586 : 59-80 (2015).

Algorithmic and experimental aspects II

- A probabilistic analysis of Sumsweep M. Borassi, P. Crescenzi, L. Trevisan : An Axiomatic and an Average-Case Analysis of Algorithms and Heuristics for Metric Properties of Graphs. SODA 2017 : 920-939
- Algorithms with BFS certificates and a deterministic analyze of Sumsweep F. F. Dragan, M. Habib, L. Viennot : Revisiting Radius, Diameter, and all Eccentricity Computation in Graphs through Certificates. CoRR abs/1803.04660 (2018).

Examples of applications

- Application to computational geometry V. Chepoi, F. Dragan, B. Estellon, M. Habib and Y. Vaxes : Diameters, centers, and approximating trees of delta-hyperbolic geodesic spaces and graphs, ACM Symposium on Computational Geometry, 2008 : 59-68.
- Application of IFUB to Facebook graph Paolo Boldi, Marco Rosa, Sebastiano Vigna : Robustness of Social Networks : Comparative Results Based on Distance Distributions. SocInfo 2011 : 8-21
- Application of Sum-sweep A. Naz, B. Piranda, J. Bourgeois, S. C. Goldstein : Electing an Approximate Center in a Huge Modular Robot with the k-BFS SumSweep Algorithm, Conference : 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018)

Complexity aspects

- M. Borassi, P. Crescenzi, M. Habib, *Into the Square : On the Complexity of Some Quadratic-time Solvable Problems*, **Electr. Notes Theor. Comput. Sci.** 322 : 51-67 (2016).
- V. Chepoi, F. Dragan, M. Habib, Y. Vaxès and H. Alrasheed, *Fast Approximation of Centrality and Distances in Hyperbolic Graphs*, **J. of Graph Algorithms and Applications** 23(2) : 393-433 (2019).
- Guillaume Ducoffe, Michel Habib, Laurent Viennot, *Fast Diameter Computation Within Split Graphs*, **COCOA** 2019 : 155-167.
- Guillaume Ducoffe, Michel Habib, Laurent Viennot : *Diameter computation on H-minor free graphs and graphs of bounded (distance) VC-dimension*, **SODA 2020**.

Many thanks for your attention ! !