Radius and diameter computations in huge graphs and some extensions,

Michel Habib
habib@liafa.univ-paris-diderot.fr
http://www.liafa.univ-paris-diderot.fr/~habib

Ecole ResCom, Furiani
May 2014

- I am playing with graph algorithms since ....

- I am playing with graph algorithms since ....
- Graphs or networks :
  A. Sainte-Laguë, *Les réseaux ou graphes*, **Gauthier-Villars**, Paris,1926.

- I am playing with graph algorithms since ....
- Graphs or networks :
  A. Sainte-Laguë, *Les réseaux ou graphes*, **Gauthier-Villars**, Paris,1926.
- One of the main problems I am looking at :
  What can you learn about the structure of a given graph using a series of consecutive graph searches ?

- I am playing with graph algorithms since ....
- Graphs or networks :
  A. Sainte-Laguë, *Les réseaux ou graphes*, **Gauthier-Villars**, Paris,1926.
- One of the main problems I am looking at :
  What can you learn about the structure of a given graph using a series of consecutive graph searches ?
- From now on :
  Graphs are undirected and supposed to be finite and connected.

Schedule of the talk

Graph searches

Schedule of the talk

Graph searches

Diameter computations

Schedule of the talk

- Graph searches
- Diameter computations
- Computing diameter using fewest BFS possible

Schedule of the talk

Graph searches

Diameter computations

Computing diameter using fewest BFS possible

The Stanford Database

Schedule of the talk

Graph searches

Diameter computations

Computing diameter using fewest BFS possible

The Stanford Database

Recents results

Schedule of the talk

- Graph searches
- Diameter computations
- Computing diameter using fewest BFS possible
- The Stanford Database
- Recents results
- Huge graphs

Schedule of the talk

Graph searches

Diameter computations

Computing diameter using fewest BFS possible

The Stanford Database

Recents results

Huge graphs

Consequences and perspectives

Graph searches are very well known and used in many situations :

1. "Fil d'ariane" in the Greek mythology.

Graph searches are very well known and used in many situations :

1. "Fil d'ariane" in the Greek mythology.
2. Euler (1735) for solving the famous walk problem in Kœnisberg city

Graph searches are very well known and used in many situations :

1. "Fil d'ariane" in the Greek mythology.
2. Euler (1735) for solving the famous walk problem in Kœnisberg city
3. Euler's theorem proved by Hierholzer in 1873.

Graph searches are very well known and used in many situations :

1. "Fil d'ariane" in the Greek mythology.
2. Euler (1735) for solving the famous walk problem in Kœnisberg city
3. Euler's theorem proved by Hierholzer in 1873.
4. Tremaux (1882) and Tarry (1895) using DFS to solve maze problems

Graph searches are very well known and used in many situations :

1. "Fil d'ariane" in the Greek mythology.
2. Euler (1735) for solving the famous walk problem in Kœnisberg city
3. Euler's theorem proved by Hierholzer in 1873.
4. Tremaux (1882) and Tarry (1895) using DFS to solve maze problems
5. Fleury, proposed a nice algorithm to compute an Euler Tour, cited in E. Lucas, Récréations mathématiques, Paris, 1891.

Graph searches are very well known and used in many situations :

1. "Fil d'ariane" in the Greek mythology.
2. Euler (1735) for solving the famous walk problem in Kœnisberg city
3. Euler's theorem proved by Hierholzer in 1873.
4. Tremaux (1882) and Tarry (1895) using DFS to solve maze problems
5. Fleury, proposed a nice algorithm to compute an Euler Tour, cited in E. Lucas, Récréations mathématiques, Paris, 1891.
6. Computer scientists from 1950, in particular in the 70's, Tarjan for new applications of DFS....

Graph searches are very well known and used in many situations :

1. "Fil d'ariane" in the Greek mythology.
2. Euler (1735) for solving the famous walk problem in Kœnisberg city
3. Euler's theorem proved by Hierholzer in 1873.
4. Tremaux (1882) and Tarry (1895) using DFS to solve maze problems
5. Fleury, proposed a nice algorithm to compute an Euler Tour, cited in E. Lucas, Récréations mathématiques, Paris, 1891.
6. Computer scientists from 1950, in particular in the 70's, Tarjan for new applications of DFS....
7. 4 points characterizations Corneil, Krueger (2008), and the definition of LDFS a new interesting basic search.

Tarry

LE PROBLÈME DES LABYRINTHES;

PAR M. G. TARRY.

Tout labyrinthe peut être parcouru en une seule course, en passant deux fois en sens contraire par chacune des allées, sans qu'il soit nécessaire d'en connaître le plan.

Pour résoudre ce problème, il suffit d'observer cette règle unique :

Ne reprendre l'allée initiale qui a conduit à un carrefour pour la première fois que lorsqu'on ne peut pas faire autrement.

Nous ferons d'abord quelques remarques.

A un moment quelconque, avant d'arriver à un car-

Umberto Eco, "Il nome della rosa", Roman, 1980

« Pour trouver la sortie d'un labyrinthe, récita en effet Guillaume, il n'y a qu'un moyen. A chaque nœud nouveau, autrement dit jamais visité avant, le parcours d'arrivée sera marqué de trois signes. Si, à cause de signes précédents sur l'un des chemins du nœud, on voit que ce nœud a déjà été visité, on placera un seul signe sur le parcours d'arrivée. Si tous les passages ont été déjà marqués, alors il faudra reprendre la même voie, en revenant en arrière. Mais si un ou deux passages du nœud sont encore sans signes, on en choisira un quelconque, pour y apposer deux signes. Quand on s'achemine par un passage qui porte un seul signe, on en appposera deux autres, de façon que ce passage en porte trois dorénavant. Toutes les parties du labyrinthe devraient avoir été parcourues si, en arrivant à un nœud, on ne prend jamais le passage avec trois signes, sauf si d'autres passages sont encore sans signes.

– Comment le savez-vous ? Vous êtes expert en labyrinthes ?

– Non, je récite un extrait d'un texte antique que j'ai lu autrefois.

– Et selon cette règle, on sort ?

– Presque jamais, que je sache. Mais nous tenterons quand même. Et puis dans les prochains jours j'aurai des verres et j'aurai le temps de mieux me pencher sur les livres. Il se peut que là où le parcours des cartouches nous embrouille, celui des livres nous donne une règle.

Some definitions

Graph Search

The graph is supposed to be connected so as the set of visited vertices. After choosing an initial vertex, a search of a connected graph visits each of the vertices and edges of the graph such that a new vertex is visited only if it is adjacent to some previously visited vertex.

At any point there may be several vertices that may possibly be visited next. To choose the next vertex we need a tie-break rule. The breadth-first search (BFS) and depth-first search (DFS) algorithms are the traditional strategies for determining the next vertex to visit.

Variations

Graph Traversal

The set of visited vertices is not supposed to be connected (used for computing connected components for example)

Variations

Graph Traversal

The set of visited vertices is not supposed to be connected (used for computing connected components for example)

Graph Searching for cops and robbers games on a graph

The name Graph searching is also used in this context, with a slightly different meaning. Relationships with width graph parameters such as treewidth.

Our main question

Main Problem

What kind of knowledge can we learn about the structure of a given graph via graph searching (i.e. with one or a series of successive graph searches) ?

Goals

Our main question

Main Problem

What kind of knowledge can we learn about the structure of a given graph via graph searching (i.e. with one or a series of successive graph searches) ?

Goals

- Building bottom up graph algorithms from well-known graph searches

Our main question

Main Problem

What kind of knowledge can we learn about the structure of a given graph via graph searching (i.e. with one or a series of successive graph searches) ?

Goals

- Building bottom up graph algorithms from well-known graph searches
- Develop basic theoretic tools for the structural analysis of graphs

Our main question

Main Problem

What kind of knowledge can we learn about the structure of a given graph via graph searching (i.e. with one or a series of successive graph searches) ?

Goals

- Building bottom up graph algorithms from well-known graph searches
- Develop basic theoretic tools for the structural analysis of graphs
- Applications on huge graphs :
  No need to store sophisticated data structures, just some labels on each vertex,

We can play with :

1. Find new uses of already known searches or describe new interesting searches designed for special purpose

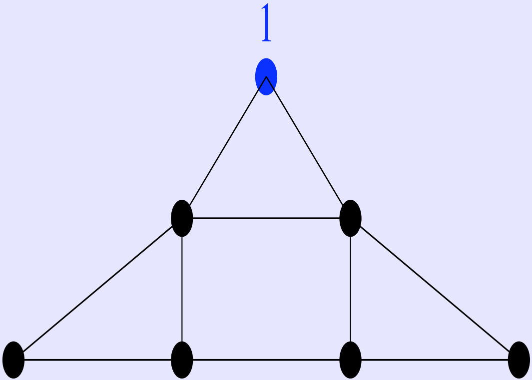
We can play with :

1. Find new uses of already known searches or describe new interesting searches designed for special purpose
2. Seminal paper :
   D.G. Corneil et R. M. Krueger, A unified view of graph searching, SIAM J. Discrete Math, 22, Num 4 (2008) 1259-1276

Basic graph searches

- Generic search, BFS, DFS
- LBFS, LDFS
- But also MNS, MCS

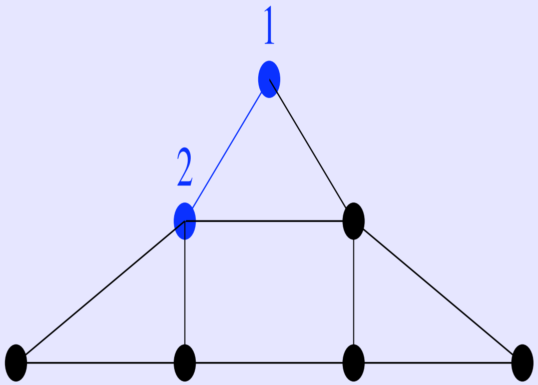
Generic Search



Invariant

At each step, an edge between a visited vertex and a unvisited one is selected

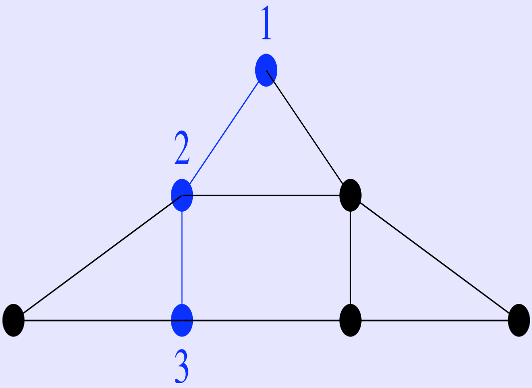
Generic Search



Invariant

At each step, an edge between a visited vertex and a unvisited one is selected

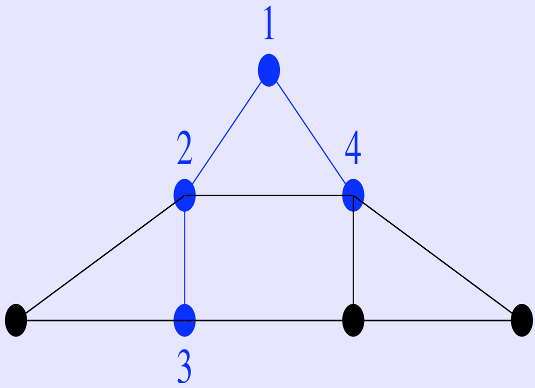
Generic Search



Invariant

At each step, an edge between a visited vertex and a unvisited one is selected

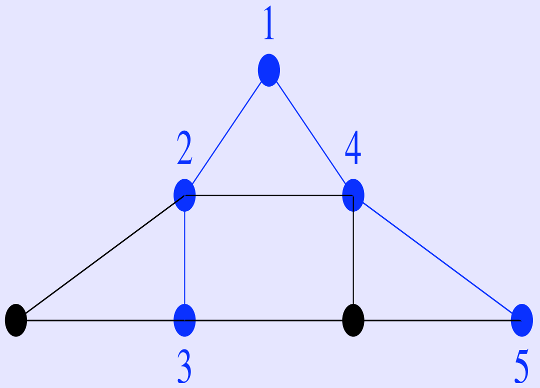
Generic Search



Invariant

At each step, an edge between a visited vertex and a unvisited one

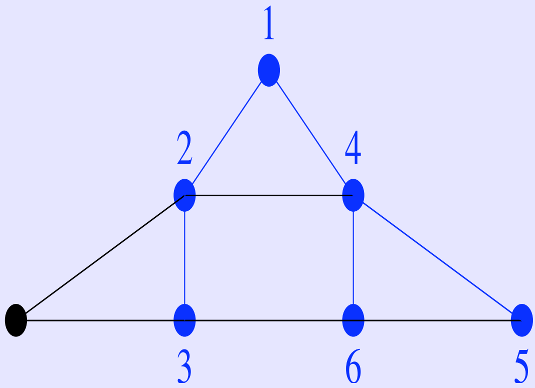
Generic Search



Invariant

At each step, an edge between a visited vertex and a unvisited one is selected

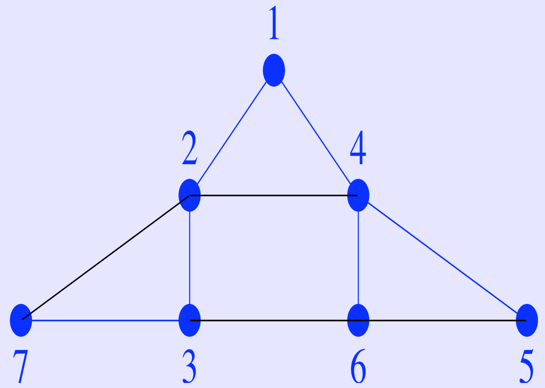
Generic Search



Invariant

At each step, an edge between a visited vertex and a unvisited one is selected

Generic Search



Invariant

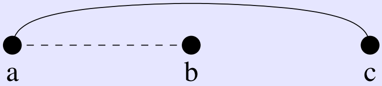
At each step, an edge between a visited vertex and a unvisited one is selected

Generic search

$S \leftarrow \{s\}$
for $i \leftarrow 1$ **to** n **do**
  Pick an unumbered vertex v of S
  $\sigma(i) \leftarrow v$
  foreach *unumbered vertex* $w \in N(v)$ **do**
    if $w \notin S$ **then**
      Add w to S
    end
  end
end

Generic question ?

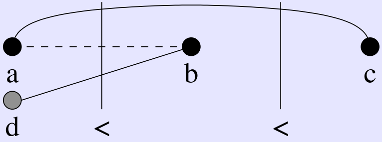
Let a, b et c be 3 vertices such that $ab \notin E$ et $ac \in E$.



Under which condition could we visit first a then b and last c ?

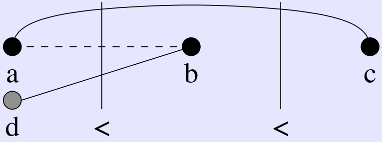
Property (Generic)

For an ordering σ on V, if $a <_\sigma b <_\sigma c$ and $ac \in E$ and $ab \notin E$, then it must exist a vertex d such that $d <_\sigma b$ et $db \in E$



Property (Generic)

For an ordering σ on V, if $a <_\sigma b <_\sigma c$ and $ac \in E$ and $ab \notin E$, then it must exist a vertex d such that $d <_\sigma b$ et $db \in E$



Theorem

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

Most of the searches that we will study are refinement of this generic search

i.e. we just add new rules to follow for the choice of the next vertex to be visited

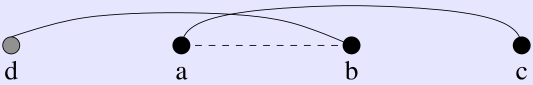
Graph searches mainly differ by the management of the tie-break set

Parcours en largeur (BFS)

Données: Un graphe $G = (V, E)$ et un sommet source s
Résultat: Un ordre total σ de V
Initialiser la file S à s
pour $i \leftarrow 1$ **à** n **faire**
    Extraire le sommet v de la tête de la file S
    $\sigma(i) \leftarrow v$
    pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**
        si w *n'est pas dans* S **alors**
            Ajouter w en fin de la file S
        fin
    fin
fin

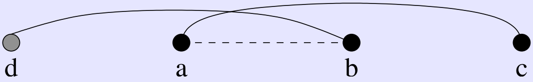
Property (BFS)

For an ordering σ on V, if $a <_\sigma b <_\sigma c$ and $ac \in E$ and $ab \notin E$, then it must exist a vertex d such that $d <_\sigma a$ et $db \in E$



Property (BFS)

For an ordering σ on V, if $a <_\sigma b <_\sigma c$ and $ac \in E$ and $ab \notin E$, then it must exist a vertex d such that $d <_\sigma a$ et $db \in E$



Theorem

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

Applications of BFS

1. Distance computations (unit length), diameter and centers
2. BFS provides a useful layered structure of the graph
3. Using BFS to search an augmenting path provides a polynomial implementation of Ford-Fulkerson maximum flow algorithm.

Recent results

- LDFS Lexicographic Depth First Search with application to hamiltonicity

Recent results

- LDFS Lexicographic Depth First Search with application to hamiltonicity
- Some other Lexicographic Searches : LexUP, LexDown

Recent results

- LDFS Lexicographic Depth First Search with application to hamiltonicity
- Some other Lexicographic Searches : LexUP, LexDown
- Recognition of cocomparability graphs using a series of LBFS

Recent results

- LDFS Lexicographic Depth First Search with application to hamiltonicity
- Some other Lexicographic Searches : LexUP, LexDown
- Recognition of cocomparability graphs using a series of LBFS
- Many questions about fixed points

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), J. Dusart (Paris), R. Charpey (Paris), M. Borassi (Firence)
and discussion with many 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).

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.

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.

- 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 $O(\frac{n^3}{log^2 n})$, 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 $O(\frac{n^3}{log^2 n})$, 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.

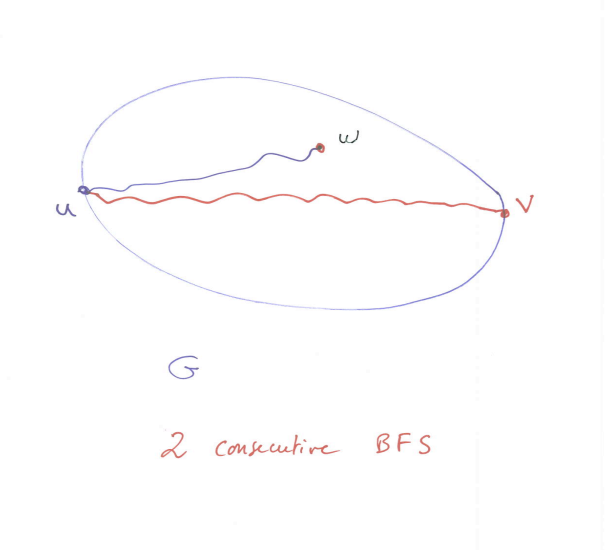
1. Let us consider the procedure called : 2 consecutive BFS [1]

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

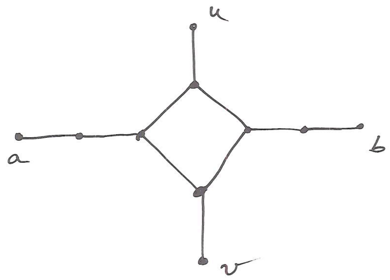
First theorem

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 $radius(G) = \lceil \frac{diam(G)}{2} \rceil$

Unfortunately it is not an algorithm !



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 partial subgraph to obtain an upper bound by computing its exact diameter in linear time.

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 partial subgraph to obtain an upper bound by computing its exact diameter in linear time.
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 .

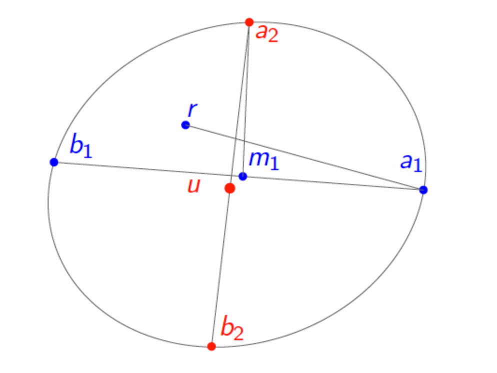
Chordal graphs

1. A graph is chordal if it has no chordless cycle of length ≥ 4 .
2. If G is a chordal graph, Corneil, Dragan, H., 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, H., 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



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

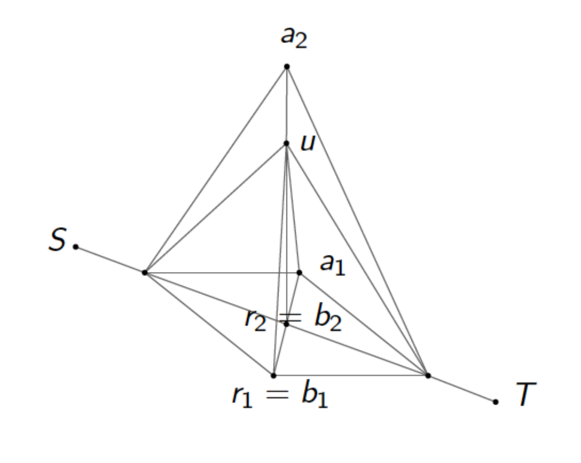
Intuition behind the 4-sweep heuristics

- 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 ! !



An exact algorithm !

Compute the excentricity of the leaves of a BFS rooted in m_1 with a stop 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

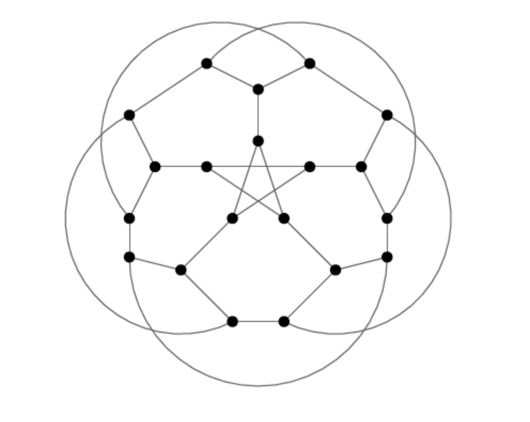
Let us consider $y \in L(j)$ with $j < i$. $\forall z \in \cup_{1 \leq k \leq i-1} L(k)$
$dist(z, y) \leq 2(i-1)$
Therefore $ecc(y) \leq ecc(x)$ or the extreme vertices from y belong to lower layers and have already been considered.

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



Results :

v	# of graphs in which v BFSes done on the average					
		Number n of vertices				
	Total	$\leq 10^3$	$\leq 10^4$	$\leq 10^5$	$\leq 10^6$	$> 10^6$
$v = 5$	29	2	8	9	10	0
$5 < v \leq 100$	123	17	44	43	11	8
$100 < v \leq 1000$	21	1	3	10	4	3
$1000 < v \leq 10^4$	18	0	4	12	1	1
$10^4 < v \leq 10^5$	8	0	0	3	3	2

- The 200th graph: Facebook network
  - 721.1M nodes and 68.7G edges
  - After 17 BFSes...

Diametre Facebook = 41 ! Backstrom, Boldi, Rosa, Uganden, Vigna 2011

Comments

- 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.

Comments

- 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.
- The 4-sweep method alway 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

- 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.
- The 4-sweep method alway 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 dependant are better that 1000 independant 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 dependant are better that 1000 independant 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 dependant are better that 1000 independant 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 to appear in 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-point condition. Which can be checked in linear time for BFS and DFS.

Graphe	$\frac{\textit{Vertices}}{\textit{Edges}}$	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

Easy extensions

1. To weighted graphs by replacing BFS with Dijkstra's algorithm

Easy extensions

1. To weighted graphs by replacing BFS with Dijkstra's algorithm
2. To directed graphs

A method symmetric for computing radius and diameter

M. Borassi, P. Crescenzi, R. Grossi, M.H., 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.

A method symmetric for computing radius and diameter

M. Borassi, P. Crescenzi, R. Grossi, M.H., 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 to k-sweep.

A method symmetric for computing radius and diameter

M. Borassi, P. Crescenzi, R. Grossi, M.H., 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 to k-sweep.
- we generalize to maintain k values in each vertex.

A method with no name yet

- 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. Increment i.
- The maximum eccentricity found, i.e. $\max_{i=1,\ldots,k} exc(v_i)$, is a lower bound for the diameter.
- Compute the eccentricity of w, the vertex minimizing $\sum_{i=1}^{k} d(w, v_i)$.
- The minimum eccentricity found, i.e. $\min\{\min_{i=1,\ldots,k} exc(v_i), exc(w)\}$, is an upper bound for the radius.

Replacing Sum by Max does not change.
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 $Diam(G) \leq \frac{2}{k}\sum_{i=1}^{k} d(v_i, y)$.

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 $Diam(G) \leq \frac{2}{k}\sum_{i=1}^{k} d(v_i, y)$.

proof

$kDiam(G) = \sum_{i=1}^{k} d(x,y) \geq \sum_{i=1}^{k} [d(x,v_i) + d(v_i,y)] = \sum_{i=1}^{k} d(x,v_i) + \sum_{i=1}^{k} d(v_i,y)$. □

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 $kRadius(G) = \sum_{i=1}^{k} d(x, y) \geq \sum_{i=1}^{k} [d(x, v_i) + d(v_i, y)] = \sum_{i=1}^{k} d(x, v_i) + \sum_{i=1}^{k} d(v_i, y)$. $\square$

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.

Real Applications

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 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

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.

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 failure is bounded by the δ-hyperbolicity of the graph.

Nice

Because many real networks have small δ-hyperbolicity.

The difficulty of the certificate

δ-hyperbolicity and treewidth (existence of big grids as subgraphs) must play a role.

1. To handle huge graphs we already have : graph searches.

1. To handle huge graphs we already have : graph searches.
2. But BFS is not so easy to program in a distributed environment

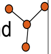
1. To handle huge graphs we already have : graph searches.
2. But BFS is not so easy to program in a distributed environment
3. For example, using Map - Reduce operations as popularized by Google.

- Some hope : Layered search is not so bad.

- Some hope : Layered search is not so bad.
- We have some theoretical results on LL

- Some hope : Layered search is not so bad.
- We have some theoretical results on LL
- We do not know if BFS is really needed ?

Restricted Families of Graphs

GRAPH CLASS	LL	LL+	BFS	LBFS	
chordal graphs	$\geq D-2$ [2] Fig. 4	$\geq D-2$ [2] Fig. 5	$\boxed{\geq D-1}$ [*] Fig. 2	$\geq D-1$ [6] Fig.6	No induced cycles of length >3
AT-free graphs	$\geq D-2$ [*] Fig. 3	$\boxed{\geq D-1}$ [*] Fig. 7	$\geq D-2$ [*] Fig. 3	$\geq D-1$ [3] Fig. 7	No asteroidal triples
{AT,claw}-free graphs	$\geq D-1$ [*] Fig. 2	$\boxed{= D}$ [*]	$\geq D-1$ [*] Fig. 2	$= D$ [*]	No asteroidal triples and 
interval graphs	$\geq D-1$ [*] Fig. 2	$\boxed{= D}$ [*]	$\geq D-1$ [*] Fig. 2	$= D$ [6]	The intersection graph of intervals of a line
hole-free graphs	$\boxed{\geq D-2}$ [*] Fig. 8	$\geq D-2$ [*] Fig. 8	$\geq D-2$ [*] Fig. 8	$\geq D-2$ [*] Fig. 8	No induced cycles of length >4

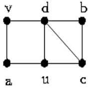


Figure 7: LBFS: u|cda|bv

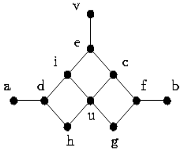
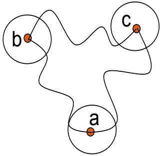


Figure 8: LBFS: u|ghic|dfe|abv



asteroidal triple a,b,c

Answers to Frequently Asked Questions

- We can really compute the exact value of the diameter for big graphs.

Answers to Frequently Asked Questions

- We can really compute the exact value of the diameter for big graphs.
- Conjecture : the computation of radius(G) and "diameter(G) within one" have the same complexity.
  "Within one" because of the case of split graphs.

- For practical algorithms the rules are not exactly the same that for classical algorithms in which only worst case complexity matters.

- For practical algorithms the rules are not exactly the same that for classical algorithms in which only worst case complexity matters.
- But we can have fun !

"While theoretical work on models of computation and methods for analyzing algorithms has had enormous payoffs, we are not done. In many situations, simple algorithms do well. We don't understand why ! Developing means for predicting the performance of algorithms and heuristics on real data and on real computers is a grand challenge in algorithms."

Challenges for Theory of Computing by :
CONDON, EDELSBRUNNER, EMERSON, FORTNOW, HABER, KARP, LEIVANT, LIPTON, LYNCH, PARBERRY, PAPADIMITRIOU, RABIN, ROSENBERG, ROYER, SAVAGE, SELMAN, SMITH, TARDOS, AND VITTER,
Report for an NSF-sponsored workshop on research in theoretical computer science.

So we need to understand

- Why the 4-sweep method works so well (analogy with Quicksort) ?

So we need to understand

- Why the 4-sweep method works so well (analogy with Quicksort) ?
- Some partial results in : Michal Parnas and Dana Ron, Testing the diameter of graphs, Random Struct. Algorithms, 2002.

So we need to understand

- Why the 4-sweep method works so well (analogy with Quicksort) ?
- Some partial results in : Michal Parnas and Dana Ron, Testing the diameter of graphs, Random Struct. Algorithms, 2002.
- For which graphs one can avoid to use $O(n)$ BFS's to compute the diameter ?

General method used so far

1. Find a linear time algorithm proved for a wide class of graphs containing trees, chordal graphs, . . .
   (Most of these algorithms are based on graph searches)

General method used so far

1. Find a linear time algorithm proved for a wide class of graphs containing trees, chordal graphs, ...
   (Most of these algorithms are based on graph searches)
2. So the study of graph classes could be useful for applications !

General method used so far

1. Find a linear time algorithm proved for a wide class of graphs containing trees, chordal graphs, . . .
   (Most of these algorithms are based on graph searches)
2. So the study of graph classes could be useful for applications !
3. Make an heuristic out of this algorithm applicable on arbitrary graphs

General method used so far

1. Find a linear time algorithm proved for a wide class of graphs containing trees, chordal graphs, . . .
   (Most of these algorithms are based on graph searches)
2. So the study of graph classes could be useful for applications !
3. Make an heuristic out of this algorithm applicable on arbitrary graphs
4. Provide certificates for partial solutions

Hints for future work

- Uses a series of graph searches for preprocessing when faced to hard combinatorial problems (already used in biological applications of interval graphs and for quadratic integer programming) Consecutive Ones property (C1P) (find a good ordering of the columns of the matrix such that in each row the ones are consecutive) A kind of filtering process !

Hints for future work

- Uses a series of graph searches for preprocessing when faced to hard combinatorial problems (already used in biological applications of interval graphs and for quadratic integer programming) Consecutive Ones property (C1P) (find a good ordering of the columns of the matrix such that in each row the ones are consecutive) A kind of filtering process !
- Develop approximation algorithms even for polynomial problems but applied on huge data on which only linear time algorithms can be processed.

- Computation of δ-hyperbolicity of graphs. A Gromov's parameter which measures the distance to a tree in a metric way. Polynomial to compute but not linearly.

- Computation of δ-hyperbolicity of graphs. A Gromov's parameter which measures the distance to a tree in a metric way. Polynomial to compute but not linearly.
- Develop similar heuristics for computing Betweenness Centrality or other centrality parameters used in biology or social networks analysis.

- Computation of δ-hyperbolicity of graphs. A Gromov's parameter which measures the distance to a tree in a metric way. Polynomial to compute but not linearly.
- Develop similar heuristics for computing Betweenness Centrality or other centrality parameters used in biology or social networks analysis.
- Community detection in networks (using LexDFS ?)

Theoretical aspects

- D. Corneil, F. Dragan, M. Habib, C. Paul, *Diameter determination on restricted families of graphs*, **Discrete Applied Mathematic**, Vol 113(2-3) : 143-166 (2001)
- V. Chepoi, F. Dragan, B. Estellon, M. Habib, Y. Vaxes, *Diameters, centers, and approximating trees of delta-hyperbolic geodesic spaces and graphs*, **ACM Symposium on Computational Geometry** 2008 : 59-68.
- V. Chepoi, F. Dragan, B. Estellon, M. Habib, Y. Vaxes, *Notes on diameters, centers, and approximating trees of δ-hyperbolic geodesic spaces and graphs*, TGCT08 Paris, **Electronic Notes in Discrete Mathematics** 31(2008)231-234.
- V. Chepoi, F. Dragan, B. Estrellon, M. Habib, Y. Vaxes et Y. Xiang *Additive Spanners and Distance and Routing Labeling Schemes for Hyperbolic Graphs*, **Algorithmica** 62-(3-4) (2012) 713-732.

Computational aspects

- C. Magnien, M. Latapy, M. Habib, *Fast computation of empirically tight bounds for the diameter of massive graphs*, **Journal of Experimental Algorithmics**, 13 (2008).
- 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).
- M. Borassi, P. Crescenzi, R. Grossi, M. Habib, W. Kosters, A. Marino and F. Takes, *On the Solvability of the Six Degrees of Kevin Bacon Game. A Faster Graph Diameter and Radius Computation Method*, accepted at **Fun with Algorithms**, june 2014

Many thanks for your attention ! !