Algorithmic in Large-scale Networks example of Compact routing

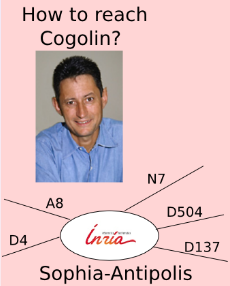
MASCOTTE

mainly related to:
DCR project
EULER project

INRIA evaluation, Paris, March 2012

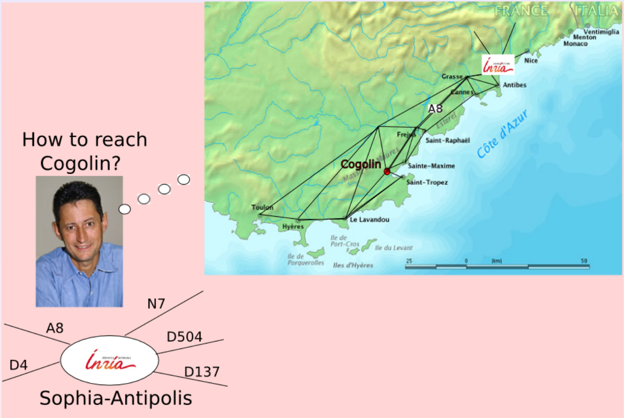
Why may Routing be difficult?

Jean-Claude wants to reach his destination



Why may Routing be difficult?

Jean-Claude wants to reach his destination

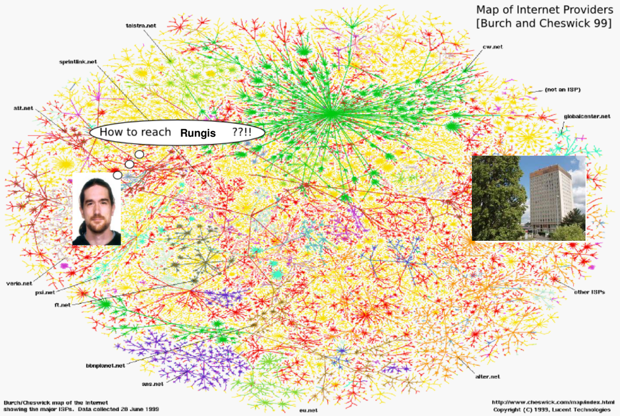


If the network is **small**, **known**, **static**, etc. Easy !!

Apply your favorite shortest path algorithm (e.g., Dijkstra)

Why may Routing be difficult?

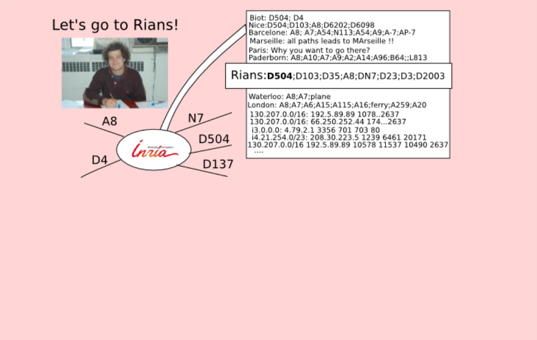
What for David?



If the network is **Huge**, **only partially known**, **dynamic**, etc.

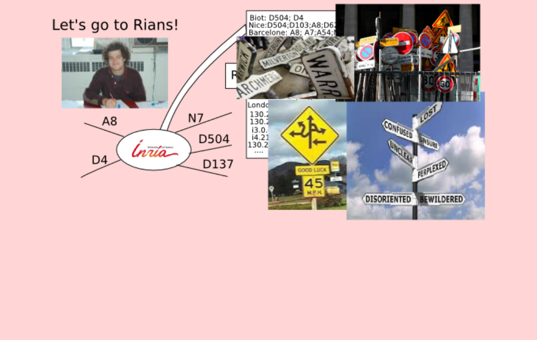
What to do ??

BGP: routing protocol of the Autonomous Systems' (AS) network



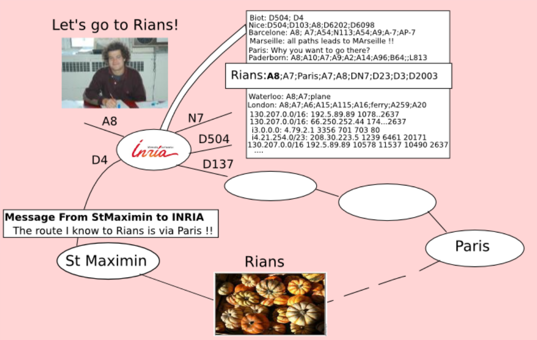
- Routing Tables (RT) attached to each AS
- 1 entry/destination: whole path stored (to avoid loops)

BGP: routing protocol of the Autonomous Systems' (AS) network



- Routing Tables (RT) attached to each AS
- 1 entry/destination: whole path stored $\Rightarrow$ huge, difficult to read

BGP: routing protocol of the Autonomous Systems' (AS) network



- Routing Tables (RT) attached to each AS
- 1 entry/destination: whole path stored $\Rightarrow$ huge, difficult to read
- to deal with dynamicity: ASs send to each other paths they know

ASs may lie (Policy), paths may be too long

Challenges

	BGP	Ideally
Routing Tables size	$O(n \log n)$ bits	$O(\log n)$ bits
Paths Length	depend on ASs policies	Shortest paths
Update Time	Long (≈ 5 min.)	Fast (using only local information)

(n is the number of ASs)

Large-scale Networks have specific structural properties

- small diameter, high clustering coefficient, power-law degree-distribution
- hyperbolicity, chordality, etc.

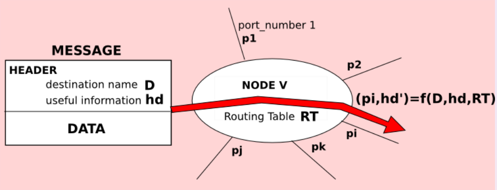
Objectives

- Understand (find new) Properties
- Use it for algorithmic purposes (not only routing)
- Model such networks
- Simulate (static/dynamic behavior)

Compact Routing

Goal To deliver a message in a distributed way

Routing Scheme protocol that directs the traffic in a network



Routing Problem: definitions of:

- Routing Tables **RT**
- Information required in the message headers **hd**
- Routing function **f**: compute next hop / may modify the header

Models

- **labelled/name independent** node Identifiers are part of the design or not
- **design port model** local labeling (port number) are part of the design or not

Performances Measures

Stretch

How far the route actually followed is from a shortest path

- **Multiplicative stretch**: $\max_{x,y \in V(G)} \leq \frac{|route(x,y)|}{dist(x,y)}$.
- **Additive stretch**: $\max_{x,y \in V(G)} \leq |route(x,y)| - dist(x,y)$.

Memory space

- Space necessary to store local routing table (per node)
- Size of the node Identifiers / message header (generally $O(\log n)$)

Time complexity

- Distributed/Centralized protocol
- Time to setup data structures
- Time to update data structures

Related Works

Names and Headers (if any) are of polylogarithmic size

network	mult. stretch	routing table	
		labelled	name-independent
arbitrary	shortest path	$O(n \log n)$ [folk]	$\Theta(n \log n)$ [Gavoille,Pérennes]
$(k \geq 2)$	$O(k)$	$O(n^{1/k})$ [Thorup,Zwick]	$\Theta(n^{1/k})$ [TZ/Abraham et al.]
trees	shortest path	$O(\log n)$ [TZ/Fraigniaud,Gavoille]	$\Omega(\sqrt{n})$ [Laing,Rajaraman]
	$2^k - 1$		$\Theta(n^{1/k})$ [Laing/Abraham et al.]
doubling-α dimension	$O(1) + \epsilon$	$O(\log \Delta)$ [Talwar/Slivkins] $O(\log n)$ [Chan et al./Abraham et al.]	$O(\epsilon^{-\alpha} \log n)$ [Abraham et al.]
planar	$1 + \epsilon$	$O(\log n)$ [Thorup]	
H-minor free	$1 + \epsilon$	$O(\|H\|! \cdot 2^{\|H\|} \log n)$ [Abraham,Gavoille]	

Related Works

Names and Headers (if any) are of polylogarithmic size

network	mult. stretch	routing table			
		labelled		name-independent	
arbitrary	shortest path	$O(n \log n)$	[folk]	$\Theta(n \log n)$	[Gavoille,Pérennes]
$(k \geq 2)$	$O(k)$	$O(n^{1/k})$	[Thorup,Zwick]	$\Theta(n^{1/k})$	[TZ/Abraham et al.]
trees	shortest path	$O(\log n)$	[TZ/Fraigniaud,Gavoille]	$\Omega(\sqrt{n})$	[Laing,Rajaraman]
	$2^k - 1$			$\Theta(n^{1/k})$	[Laing/Abraham et al.]
doubling-α dimension	$O(1) + \epsilon$	$O(\log \Delta)$	[Talwar/Slivkins]	$O(\epsilon^{-\alpha} \log n)$	[Abraham et al.]
		$O(\log n)$	[Chan et al./Abraham et al.]		
planar	$1 + \epsilon$	$O(\log n)$	[Thorup]		
H-minor free	$1 + \epsilon$	$O(\lvert H \rvert! \cdot 2^{\lvert H \rvert} \log n)$	[Abraham,Gavoille]		

In general graphs, $\Theta(n \log n)$ is optimal.

Can we do better than BGP?

Yes !! (we hope), using structural properties

Chordality

Large-scale Networks $\Rightarrow$ high clustering coefficient

$\approx$ 2 neighbors of a same node are neighbors with high probability

$\approx$ friends of my friend are my friend

$\Rightarrow$ **few large induced cycles**

Chordality

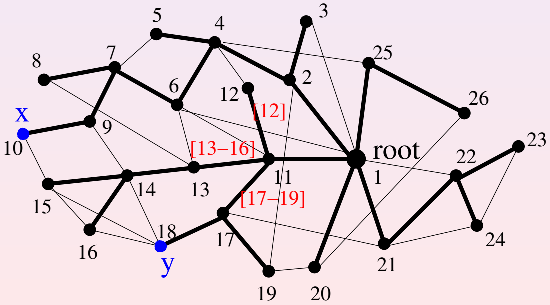
Induce cycle = cycle without chord

chordality of a graph G = length of **largest induced cycle in** G

k-chordal graph: chordality $\leq k$

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

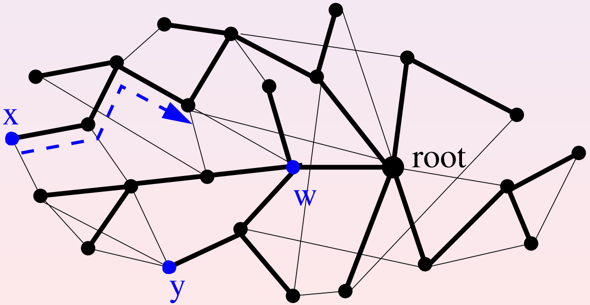


Simple Routing scheme in k-chordal graphs

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

If $x = y$, stop.
If there is $w \in N_G(x)$, an ancestor of y in T,
choose w minimizing $d_T(w, y)$;
Otherwise, choose the parent of x in T.

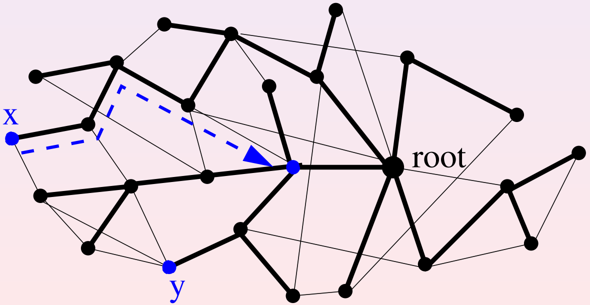


Simple Routing scheme in k-chordal graphs

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

If $x = y$, stop.
If there is $w \in N_G(x)$, an ancestor of y in T,
choose w minimizing $d_T(w, y)$;
Otherwise, choose the parent of x in T.

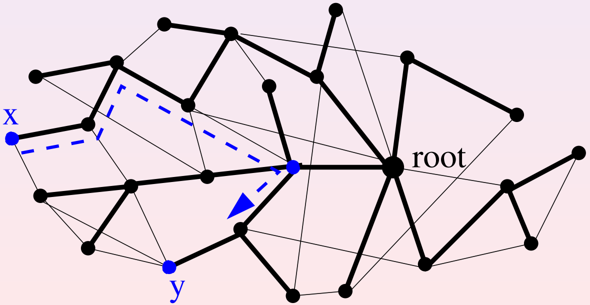


Simple Routing scheme in k-chordal graphs

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

If $x = y$, stop.
If there is $w \in N_G(x)$, an ancestor of y in T,
  choose w minimizing $d_T(w, y)$;
Otherwise, choose the parent of x in T.

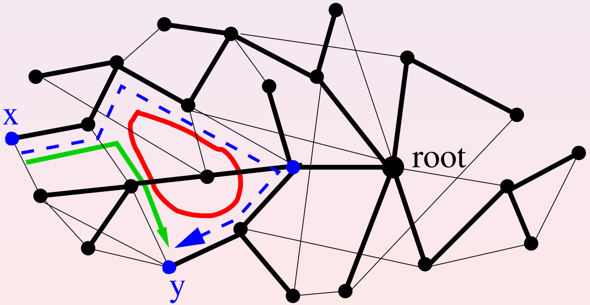


Simple Routing scheme in k-chordal graphs

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G prefix order labeling
x a source node and y a destination node

If $x = y$, stop.
If there is $w \in N_G(x)$, an ancestor of y in T,
choose w minimizing $d_T(w, y)$;
Otherwise, choose the parent of x in T.



Simple Routing scheme in k-chordal graphs

Universal, Labelled scheme, no header

G a network and T a rooted spanning tree of G — prefix order labeling

x a source node and y a destination node

> If $x = y$, stop.
> If there is $w \in N_G(x)$, an ancestor of y in T,
> choose w minimizing $d_T(w, y)$;
> Otherwise, choose the parent of x in T.

Once T has been chosen

Space: *labeling of nodes*: any rooted subtree $\Leftrightarrow$ interval
routing table: each node knows the interval of its neighbors
$O(\Delta \log n)$ bits per node

Time: easy to compute in time $O(D)$ in synchronous distributed way

Stretch: if T is a BFS-tree in a k-chordal graph: additive stretch $\leq k - 1$

In k-chordal Graphs

Names and Headers (if any) are of polylogarithmic size

network	stretch	table	computation time	
3-chordal	$+2$	$O(\frac{\log^3 n}{\log \log n})$	$O(m + n \log^2 n)$	[Dourisboure,Gavoille'02]
	$+1$	$O(\Delta \log n)$	$O(n)$	[NRS'09]
	distributed protocol / no header			
k-chordal	$k+1$	$O(k \log^2 n)$	$poly(n)$	[Dourisboure'05]
	header never changes			
	$k-1$	$O(\Delta \log n)$	$O(D)$	[NRS'09]
	distributed protocol / no header			
	$O(k \log \Delta)$	$O(k \log n)$	$O(m^2)$	[KLNS'12]

The technical slide

Theorem: Compact Routing scheme for [KLNS'12]

n-node m-edge graph with max degree Δ and using k-good decomposition

- routing tables of size $O(\max\{k \log \Delta, \log n\})$ bits computable in time $O(m^2)$.
- additive stretch is $O(k \log \Delta)$.

"History" and techniques

Cops and Robber
$k-1$ cops can capture a robber in k-chordal graphs
$\Rightarrow$ nice structural results based on separators

k- good decomposition
tree-decomposition with all bags are k-*caterpillars*

$O(m^2)$ algorithm that

1. either computes a k-good decomposition,
2. or finds an induced cycle of length $> k$

In Case 1 $\Rightarrow$ treewidth $\leq k\Delta$; treelength $\leq k-1$; hyperbolicity $\leq \frac{3}{2}k$.

Compact Routing
Combining a BFS-tree and a k-good decomposition

Further Works

short term goals:

Implement the decomposition-algorithm and simulate on large scale models

What if other properties are included?

and then?

Structural Properties of Large-scale Networks

Efficient algorithms for $\geq 10^5$-node graphs

(e.g., for hyperbolicity $O(n^4)$-algorithms cannot work)

what about NP-hard problems? (e.g., chordality)

other "hidden" properties?

Distributed algorithms

How facing dynamicity?

models of dynamicity

localized algorithms