Harvesting and analyzing large-scale directed networks: theory and practice

Sebastiano Vigna
(with Paolo Boldi, Andrea Marino and Massimo Santini)

PART I:

BUbiNG

Why a new crawler?

- Not so many open-source crawlers
- Not so configurable
- Not so extensible
- Not distributed
- NIH

Previous work

- Mercator (Najork *et al.*)
- UbiCrawler (Boldi *et al.*)
- IRLBot (WWW 2008)
- Heritrix (Internet Archive)
- Nutch (based on Hadoop)
- Bixo (based on Hadoop)
- Surprisingly little performance data

Challenges

- Use massive memory and multiple cores efficiently (does not work on a mobile phone)
- Fill bandwidth in spite of politeness (both at host and IP level)
- Stoppable/restartable
- Completely configurable
- Extensible will little effort

High Parallelism

- We use massively multiple (like 5000) fetching threads
- Every thread handles a request and is I/O bound
- Parallel threads parse and store pages
- Slow data structures are sandwiched between *lock-free* queues

Fully Distributed

- We use JGroups to set up a view on a set of agents
- Hosts are assigned to agent using consistent hashing
- URLs for which an agent is not responsible are quickly delivered to the right agent
- We use JAI4J, a thin layer over JGroups that handles job assignment.

Near–Duplicates

- We detect (presently) near-duplicates using a fingerprint of a stripped page (stored in a Bloom filter)
- The stripping includes eliminating almost all tag attributes and numbers from text
- We are going to experiment with more sophisticated methods like SimHash
- Suggestions for heuristics are welcome
- We would like to have a test collection

Fast?

- *In vitro*: >9000 pages/s average, peaks at 18000 pages/s
- Actual crawl of the .it domain done at iStella: >5000 pages/s average (single crawler), but we saturated a 2Gb/s link, so we don't really know
- ClueWeb09 (Nutch): 4.3 pages/s
- ClueWeb12 (Heritrix): 60 pages/s
- IRLbot: 1790 pages/s (unverifiable)

Almost everything broke down

- Unfortunately, when you develop on the edge...
- Hardware breaks down: €40,000 server replaced for no charge with a €60,000 server
- OS breaks down: Linux kernel's bug 862758
- JVM breaks down: try opening 5000 random-access files
- Dozens of bug reports and improvements to a number of open-source projects, including the Jericho HTML parser, Apache Software Foundation's HTTP Client, etc.

PART II:

Axioms

What is centrality?

- To understand centrality, we need mathematical properties
- One of the major goals of our research in NADINE was to isolate such properties
- In "Axioms for Centrality" we propose three properties
- (Beside sorting out the zoo of centralities)
- Do we understand centrality?

Hollywood: PageRank

Ron Jeremy

Adolf Hitler

Lloyd Kaufman

George W. Bush

Ronald Reagan

Bill Clinton

Martin Sheen

Debbie Rochon

Hollywood: Degree

William Shatner

Bess Flowers

Martin Sheen

Ronald Reagan

George Clooney

Samuel Jackson

Robin Williams

Tom Hanks

Hollywood: Betweenness

Adolf Hitler

Lloyd Kaufman

Ron Jeremy

Tony Robinson

Olu Jacobs

Max von Sydow

Udo Kier

George W. Bush

Hollywood: Katz

William Shatner

Martin Sheen

Tom Hanks

Robin Williams

George Clooney

Ronald Reagan

Bruce Willis

Samuel Jackson

Hollywood: Closeness

Lina Tjeng

Anh Loan Nguyen Thi

Ryan [illegible]oto

Chad Reed

have largest centrality

Isolated nodes ha

Bjorn van Wenum

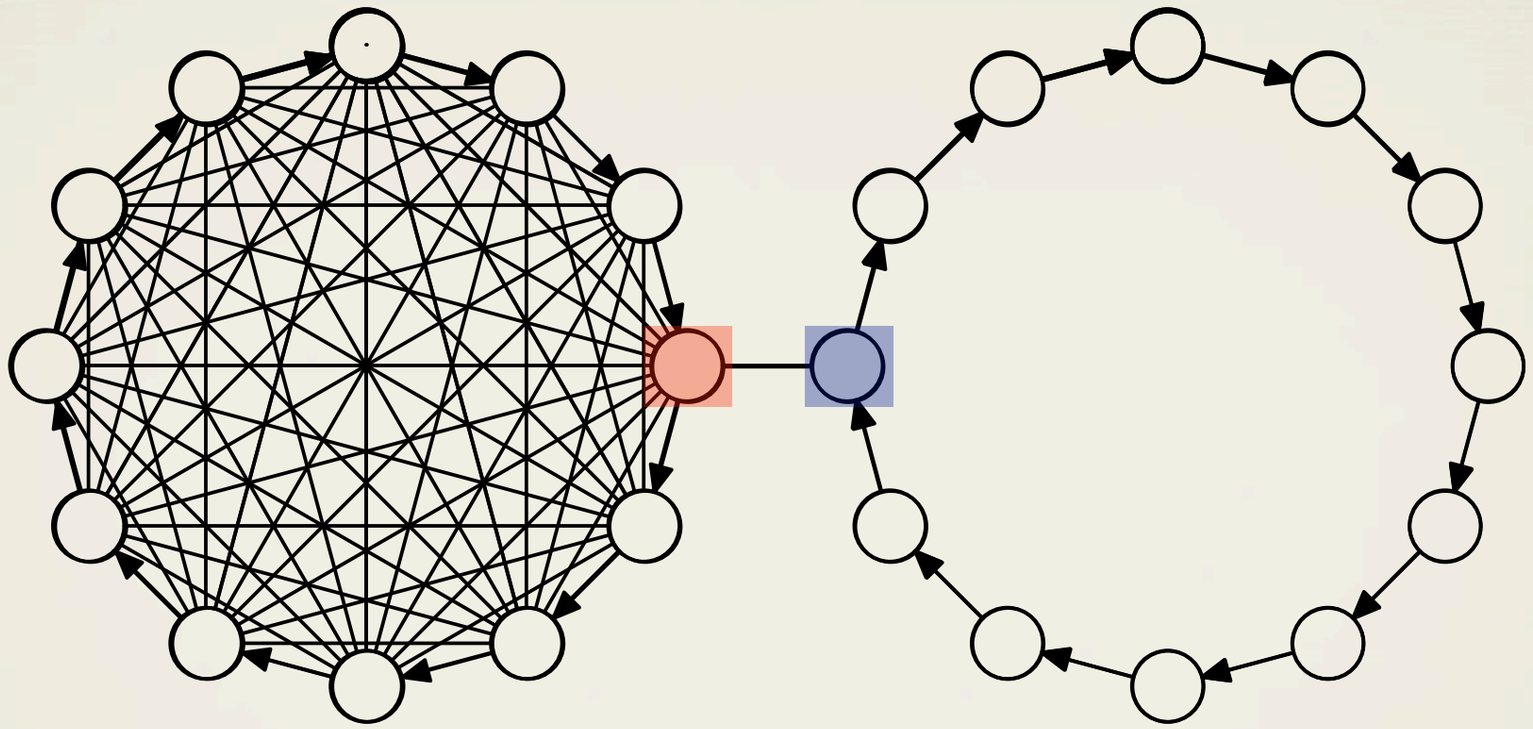
J.P. Ramackers

Herbert Sydney

R.D. Nicholson

Axiomatic

Sensitivity to density



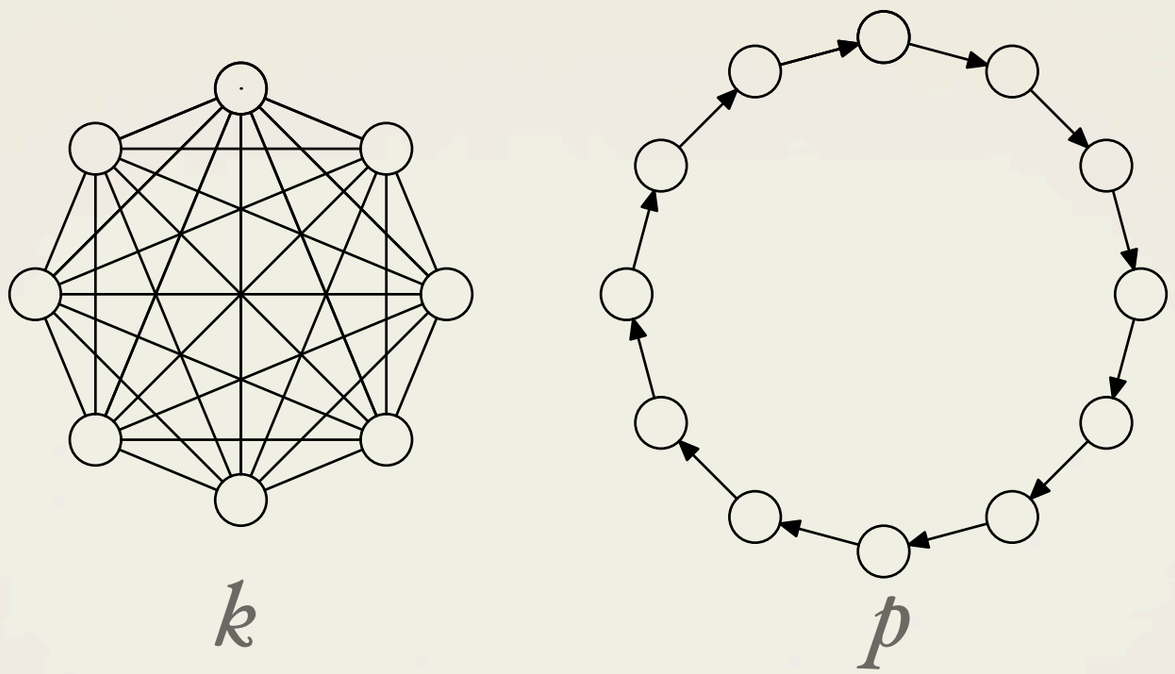
Densifying the left and the right, we expect the same node to become more important (the two trains have the same size!)

The left and the right, we expect the same node to become more important (the two trains have the same size!)

Axiomatic

Sensitivity to size

Two disjoint (or very far) components of a single network



k *p*

When *k* or *p* goes to ∞, the nodes of the corresponding subnetwork must become more important

Axiomatic

Monotonicity

When we add an arc from x to y, the score of y must increase

An axiomatic slaughter

	Density	Size	Monotonicity
Degree	yes	only k	yes
Betweenness	no (!)	only p	no
Katz	yes	only k	yes
Closeness	no	no (!)	no
Lin	no	only k	no
Harmonic	**yes**	**yes**	**yes**
PageRank	yes	no	yes
Seeley	yes	no	no
HITS	yes	only k	no
SALSA	yes	no	no
Dominant	yes	only k	no

A better closeness

- Give a warm welcome to *harmonic centrality*:

$$c_{\mathrm{harm}}(x) = \sum_{y \neq x} \frac{1}{d(y,x)}$$

- The denormalized reciprocal of the harmonic mean of all distances (even ∞)
- Inspired by the use the the harmonic mean in (Marchiori, Latora 2000)
- Quoted for undirected graphs in Tore Opsahl's blog

Hollywood: **Harmonic**

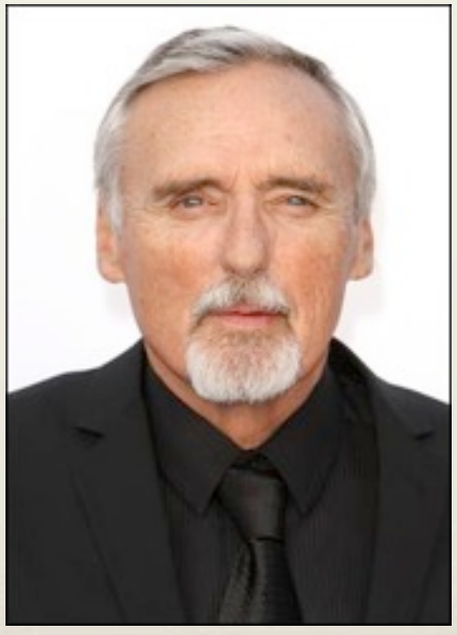
George Clooney

Samuel Jackson

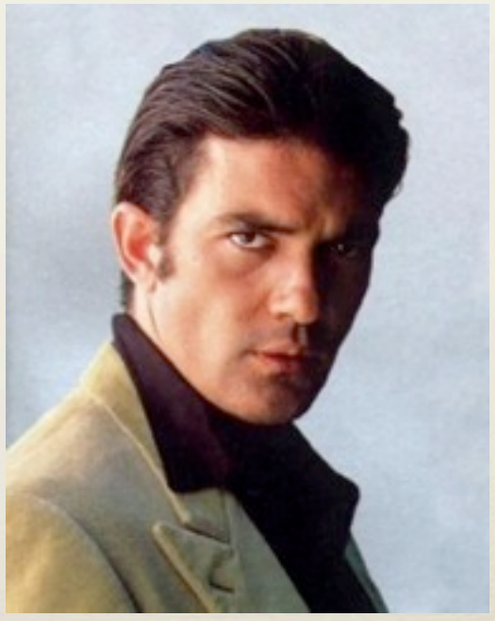
Sharon Stone

Tom Hanks

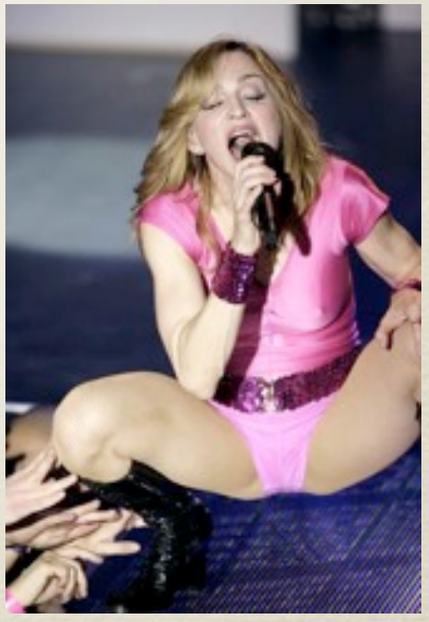
Martin Sheen



Dennis Hopper



Antonio Banderas



Madonna

People seem to like it

- Aaron Clauset is already teaching our axioms and harmonic centrality at Santa Fe Institute!
- Luca Aiello @Yahoo! Barcelona is using harmonic centrality for recommendation systems
- Spread the word!

PART III:

Algorithms

Nice idea but...

- ...computing harmonic centrality is not so easy
- In particular on directed network
- General problem of anything based on shortest paths
- Solution: approximated/probabilistic/ Monte–Carlo algorithms
- HyperBall (from Flajolet's HyperLogLog probabilistic counters)

For real

- Highly scalable, massively parallel computation
- Open-source software part of the WebGraph framework
- Run on Facebook (whole graph) using just a workstation (72GiB RAM)

Intermediate step

- For each node, we compute in sequence the number of nodes at distance exactly *t*
- Adding up over all nodes, we get the distance distribution (modulo normalization)
- Centralities can be rewritten, e.g., harmonic:

$$\sum_{t>0} \frac{1}{t} \left| \{ y \mid d(y,x) = t \} \right|$$

Dynamic Programming

- Basic idea: Palmer *et. al*, KDD '02
- Let $B_t(x)$ be the ball of radius t around x (nodes at distance at most t from x)
- Clearly $B_0(x)=\{x\}$
- But also $B_{t+1}(x)=\cup_{x\rightarrow y}B_t(y)\cup\{x\}$
- So we can compute balls by enumerating the arcs $x\rightarrow y$ and performing set unions...
- *...using a probabilistic representation!*

Performance

- On a 177K nodes / 2B arcs graph, RSD ~14%:
- Hadoop: 2875s per iteration [Kang, Papadimitriou, Sun and H. Tong, 2011]
- HyperBall on this laptop: 70s per iteration
- On a 32-core workstation: 23s per iteration
- On ClueWeb09 (4.8G nodes, 8G arcs) on a 40-core workstation: 141m (avg. 40s per iteration)

People seem to like it

- École Polytechnique Fédéral de Lausanne's x-stream project implemented HyperBall (http://labos.epfl.ch/x-stream)
- Tokyo Institute of Technology's ScaleGraph project, too (http://scalegraph.sourceforge.net/web/)
- [Named HyperANF at that time!]

Thanks!

Questions?