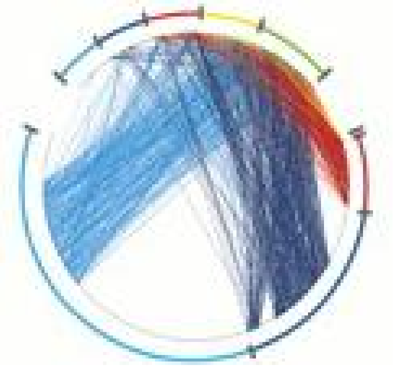
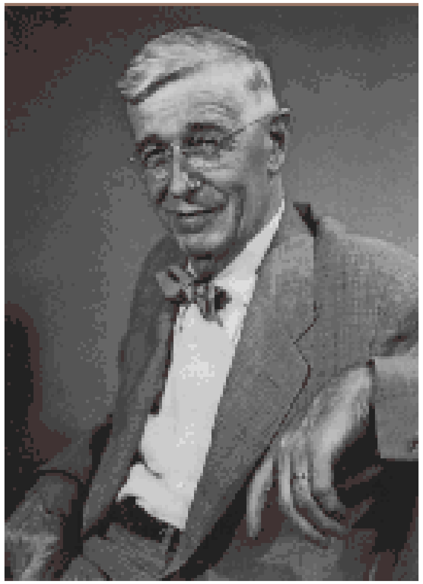
Models and Algorithms for Complex Networks

The Web graph



The history of the Web



Vannevar Bush – “As we may think” (1945)

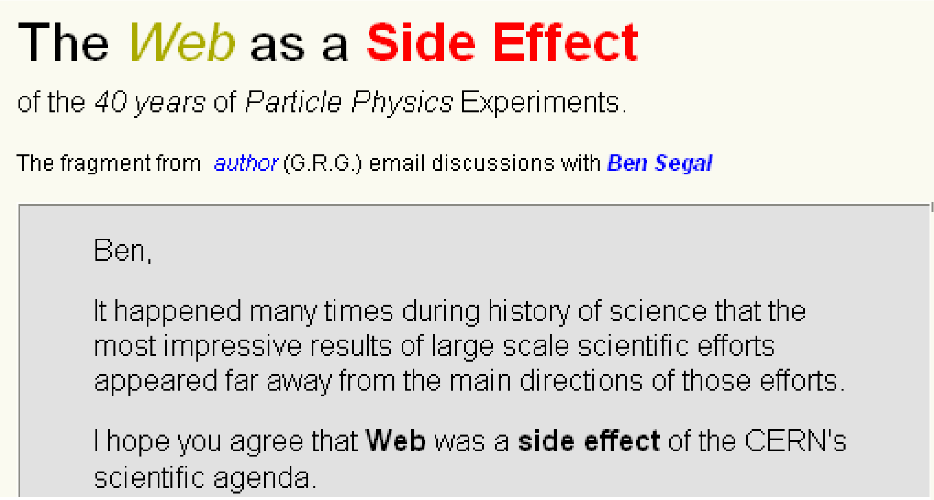
The “MEMEX”: A photo-electrical-mechanical device that stores documents and images and allows to create and follow links between them

The history of the Web

Tim Berners-Lee

1980 – CERN: Writes a notebook program
"Enquire-upon-within-everything"
that allows links to be made between
arbitrary nodes
1989 – CERN: Circulates the document
"Information management: a proposal"
1990 – CERN: The first Web browser, and the first
Web server-client communication
1994 : The creation of the WWW consortium (W3C)

The history of the Web



The *Web* as a **Side Effect**

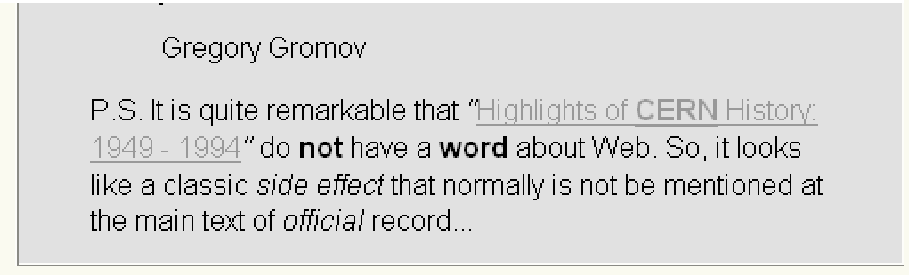
of the *40 years* of *Particle Physics* Experiments.

The fragment from *author* (G.R.G.) email discussions with ***Ben Segal***

Ben,

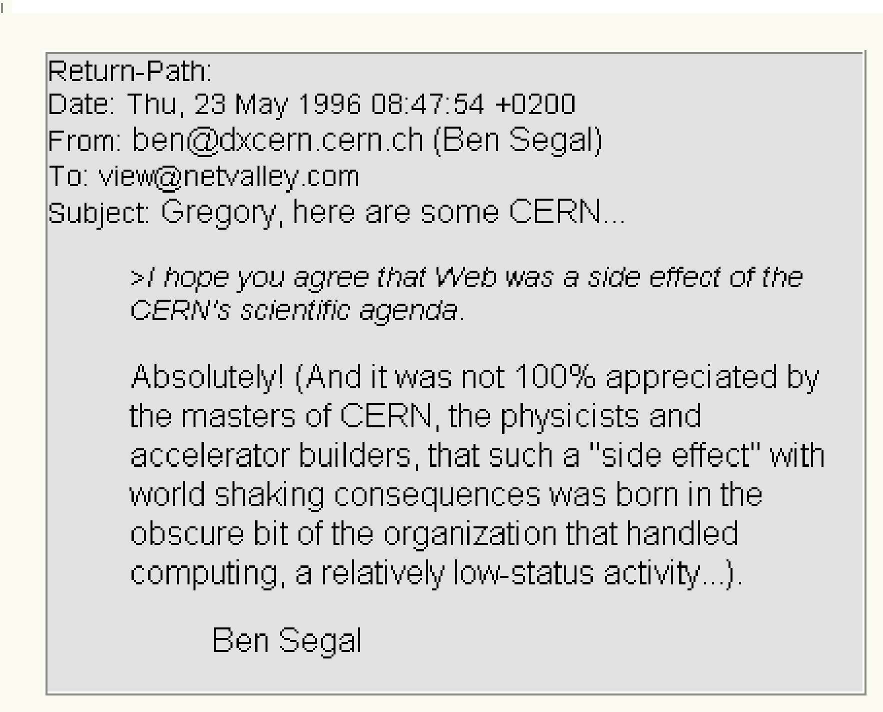
It happened many times during history of science that the most impressive results of large scale scientific efforts appeared far away from the main directions of those efforts.

I hope you agree that **Web** was a **side effect** of the CERN's scientific agenda.



Gregory Gromov

P.S. It is quite remarkable that "Highlights of **CERN** History: 1949 - 1994" do **not** have a **word** about Web. So, it looks like a classic *side effect* that normally is not be mentioned at the main text of *official* record...



Return-Path:
Date: Thu, 23 May 1996 08:47:54 +0200
From: ben@dxcern.cern.ch (Ben Segal)
To: view@netvalley.com
Subject: Gregory, here are some CERN...

>I hope you agree that Web was a side effect of the CERN's scientific agenda.

Absolutely! (And it was not 100% appreciated by the masters of CERN, the physicists and accelerator builders, that such a "side effect" with world shaking consequences was born in the obscure bit of the organization that handled computing, a relatively low-status activity...).

Ben Segal

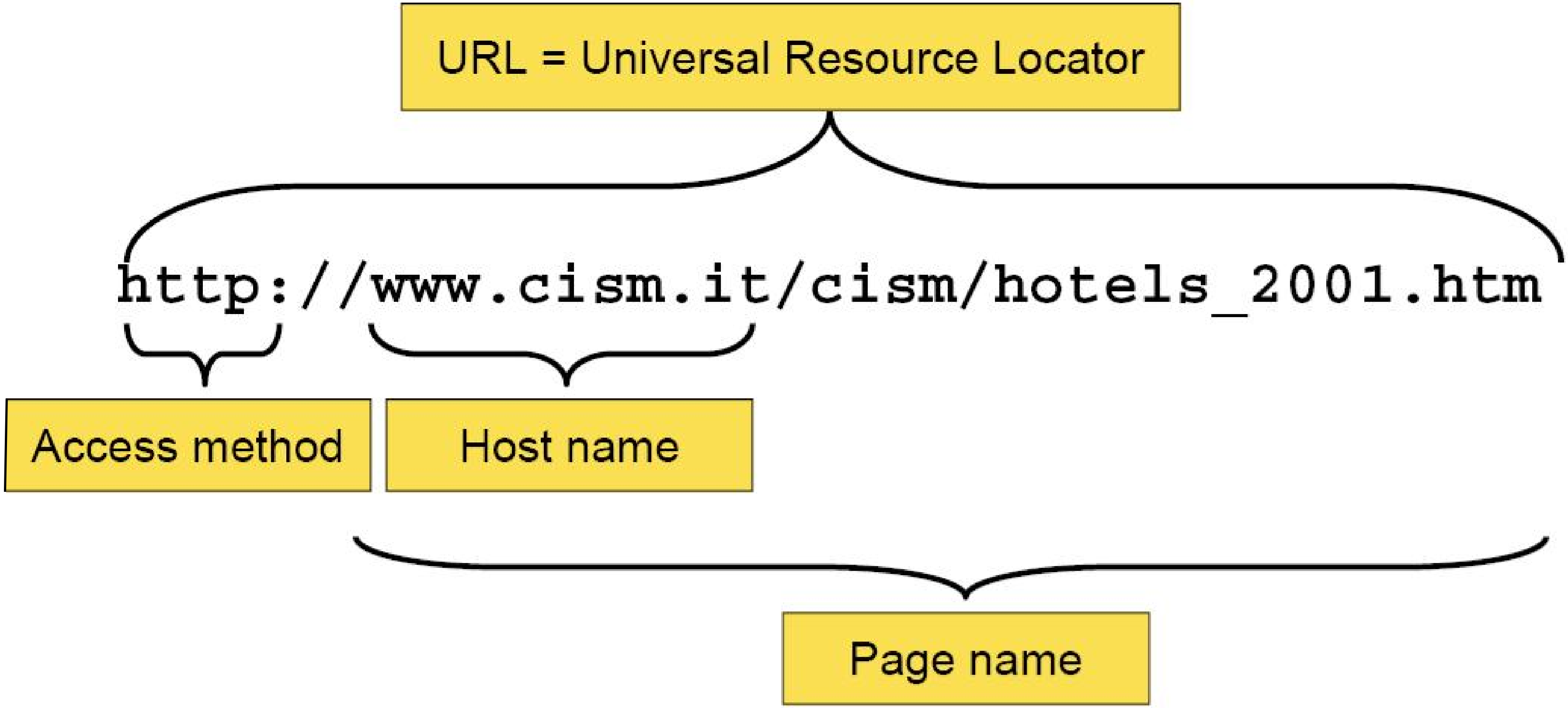
The history of the Web

Hypertext 1991: Tim Berners-Lee paper on WWW
was accepted only as a poster

Today

- The Web consists of hundreds of billions of pages
- It is considered one of the biggest revolutions in recent human history

Web page



URL = Universal Resource Locator

`http://www.cism.it/cism/hotels_2001.htm`

Access method

Host name

Page name

Which pages do we care for?

- We want to avoid “dynamic” pages
  - catalogs
  - pages generated by queries
  - pages generated by cgi-scripts (the nostradamus effect)

- We are only interested in “static” web pages

The Static Public Web

- Static
  - not the result of a cgi-bin scripts
  - no “?” in the URL
  - doesn’t change very often
  - etc.
- Public
  - no password required
  - no robots.txt exclusion
  - no “noindex” meta tag
  - etc.
- These rules can still be fooled
  - “Dynamic pages” appear static
    - browseable catalogs (Hierarchy built from DB)
  - Spider traps -- infinite url descent
    - www.x.com/home/home/home/…./home/home.html
  - Spammer games

The Web graph

- A graph G = (V, E) is defined by
  - a set V of vertices (nodes)
  - a set E of edges (links) = pairs of nodes

- The Web page graph (directed)
  - V is the set of static public pages
  - E is the set of static hyperlinks

- Many more graphs can be defined
  - The host graph
  - The co-citation graph
  - etc

Why do we care about the Web graph?

- It is the largest human artifact ever created
- Exploit the Web structure for
  - crawlers
  - search and link analysis ranking
  - spam detection
  - community discovery
  - classification/organization
- Predict the Web future
  - mathematical models
  - algorithm analysis
  - sociological understanding

The first question: what is the size of the Web?

- Surprisingly hard to answer
- Naïve solution: keep crawling until the whole graph has been explored
- Extremely simple but wrong solution: crawling is complicated because the web is complicated
  - spamming
  - duplicates
  - mirrors
- Simple example of a complication: Soft 404
  - When a page does not exists, the server is supposed to return an error code = “404”
  - Many servers do not return an error code, but keep the visitor on site, or simply send him to the home page

A sampling approach

- Sample pages uniformly at random
- Compute the percentage p of the pages that belong to a search engine repository (search engine coverage)
- Estimate the size of the Web

size(Web) = size(Search Engine)/p

- Problems:
  - how do you sample a page uniformly at random?
  - how do you test if a page is indexed by a search engine?

Sampling pages [Lawrence et al]

- Create IP addresses uniformly at random
  - problems with virtual hosting, spamming

Near uniform sampling [Henzinger et al]

- Starting from a subset of pages perform a random walk on the graph (with restarts). After “enough” steps you should end up in a random page.
  - problem: pages with high degree are more likely to be sampled

Near uniform sampling [Henzinger et al]

- Perform a random walk to obtain a random crawl. Then sample a subset of these pages

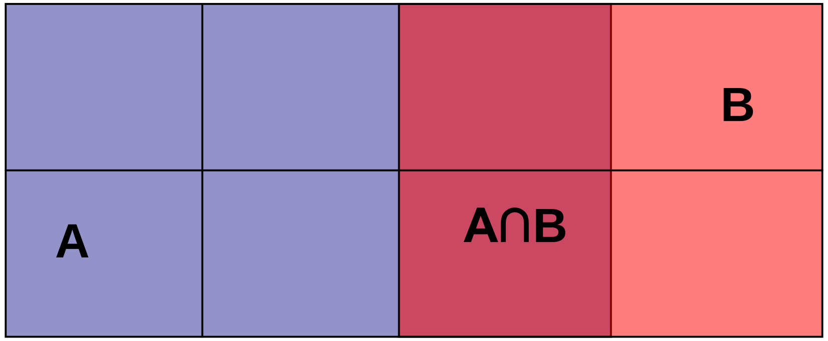
- How to sample?

$$P(X \text{ sampled}) = P(X \text{ sampled} \mid X \text{ crawled}) * P(X \text{ crawled})$$

  - sample a page with probability inversely proportional to the P(X crawled)
- Estimating P(X crawled)
  - using the number of visits in the random walk
  - using the PageRank value of the node in the crawl

Estimating the size of the indexed web

- Estimating the relative size of search engines



$\mathrm{Prob}(A \cap B | A) = |A \cap B| / |A|$ $\mathrm{Prob}(A \cap B | B) = |A \cap B| / |B|$

$|A| / |B| = \mathrm{Prob}(A \cap B | B) \,/\, \mathrm{Prob}(A \cap B | A)$

- Sample from A and compute the fraction f_1 of pages in intersection
- Sample from B and compute the fraction f_2 of pages in intersection
- Ratio f_2/f_1 is the ratio of size of A over size of B

Sampling and Checking [Bharat and Broder]

- We need to procedures:
  - Sampling procedure for obtaining a uniformly random page of a search engine
  - Checking procedure to test if a sampled page is contained in another search engine.

Sampling procedure [Bharat and Broder]

- From a collection of Web documents construct a lexicon
- Use combination of keywords to perform OR and AND queries
- Sample from the top-100 pages returned

- Biases:
  - query bias, towards rich in content pages
  - ranking bias, towards highly ranked pages

Checking procedure

- Create a strong query, with the k most significant terms
  - significance is inversely proportional to the frequency in the lexicon

- Query search engine and check if it contains a given URL
  - full URL check
  - text similarity

Results [Gulli, Signorini 2005]

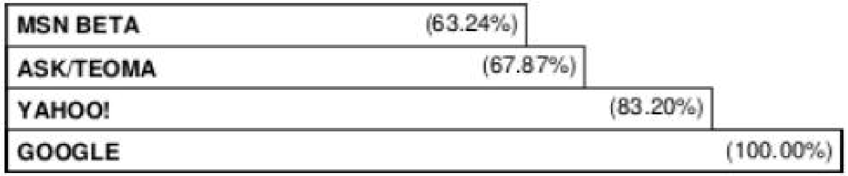


Figure 3: Estimated relative size per search engine

Figure 4 graphically represents the percentage of the indexable web that lies in each search engine's index and in their respective intersections.

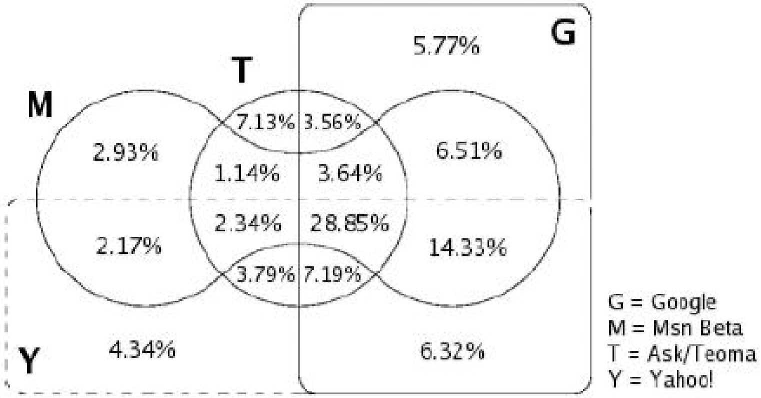


Figure 4: Results distribution across engines.

Estimating Web size

- Results indicate that the search engines are independent
  - $\text{Prob}(A \cap B | A) \approx \text{Prob}(A \cap C | C)$
  - $\text{Prob}(A \cap B | A) \approx \text{Prob}(B)$
  - if we know the size of B we can estimate the size of the Web
- In 2005: 11.5 billion

Measuring the Web

- It is clear that the Web that we see is what the crawler discovers

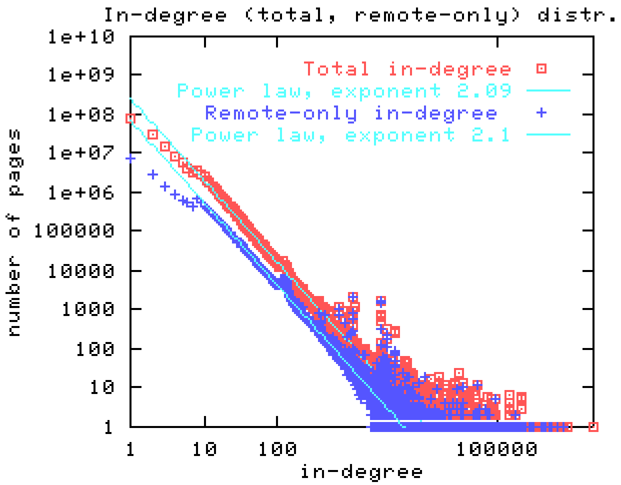
- We need large crawls in order to make meaningful measurements
- The measurements are still biased by
  - the crawling policy
  - size limitations of the crawl
  - Perturbations of the "natural" process of birth and death of nodes and links

Measures on the Web graph [Broder et al]

- Degree distributions
- Reachability
- The global picture
  - what does the Web look from far?
- Connected components
- Community structure
- The finer picture

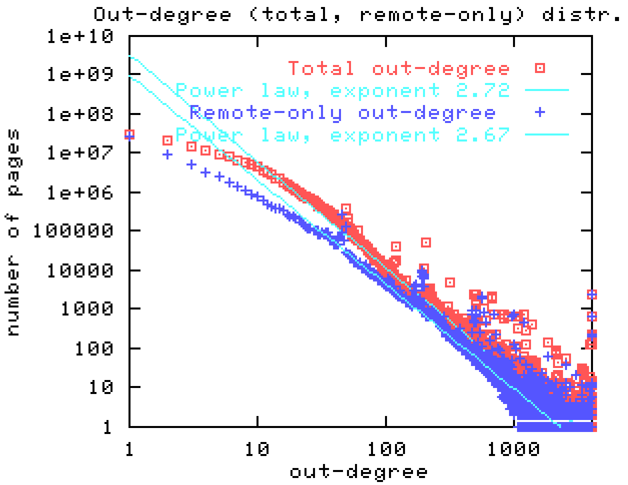
In-degree distribution

- Power-law distribution with exponent 2.1



Out-degree distribution

- Power-law distribution with exponent 2.7



The good news

- The fact that the exponent is greater than 2 implies that the expected value of the degree is a constant (not growing with n)

- Therefore, the expected number of edges is linear in the number of nodes n

- This is good news, since we cannot handle anything more than linear

Is the Web a small world?

- Based on a simple model, [Barabasi *et*. *al*.] predicted that most pages are within 19 links of each other. Justified the model by crawling nd.edu (1999)
- Well, not really!

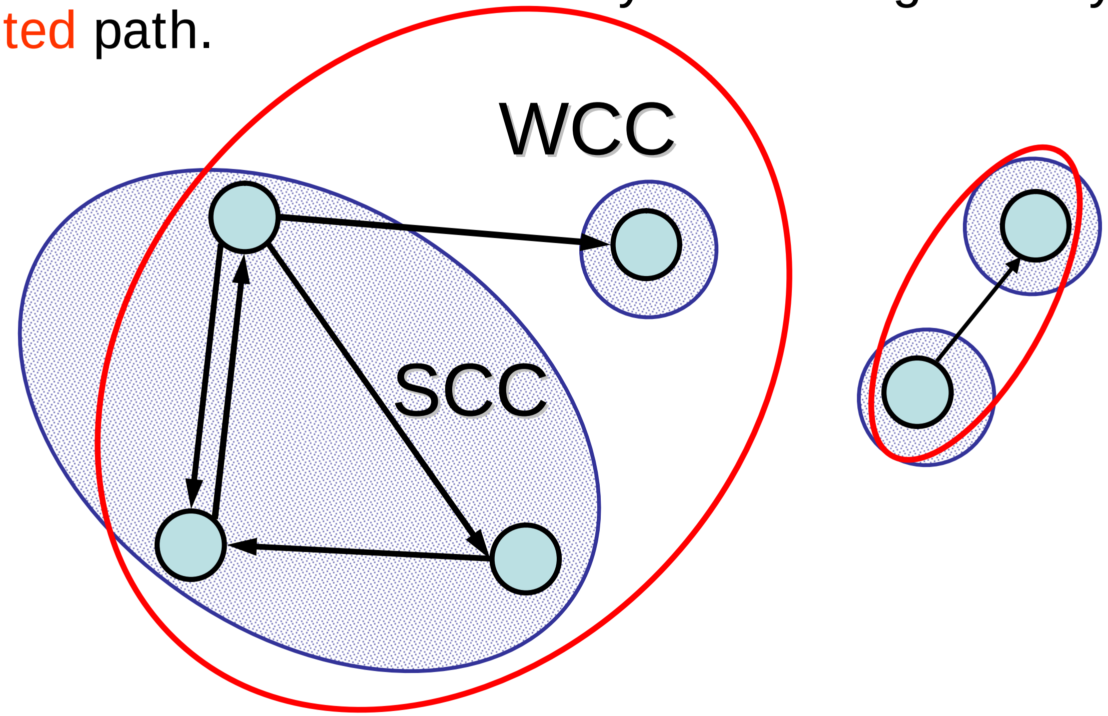
Distance measurements

- The probability that there exists a directed path between two nodes is ~25%
  - Therefore, for ~75% of the nodes there exists no path that connects them

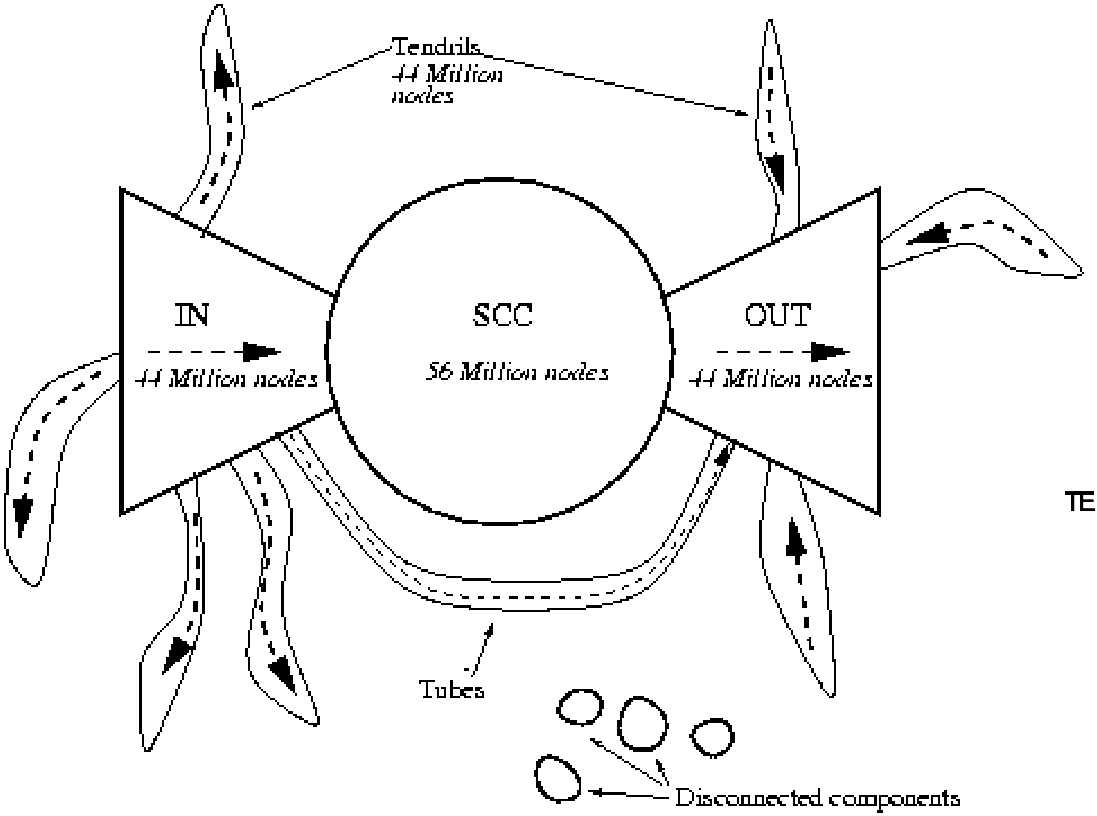
- Average directed distance between two nodes in the CORE: ~16
- Average undirected distance between two nodes in the CORE: ~7
- Maximum directed distance between two nodes in the CORE: >28
- Maximum directed distance between any two nodes in the graph: > 900

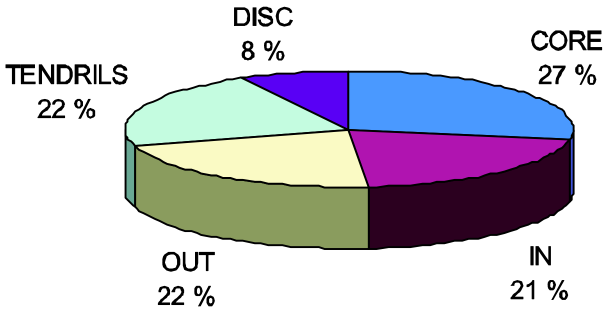
Connected components – definitions

- Weakly connected components (WCC)
  - Set of nodes such that from any node can go to any node via an undirected path
- Strongly connected components (SCC)
  - Set of nodes such that from any node can go to any node via a directed path.



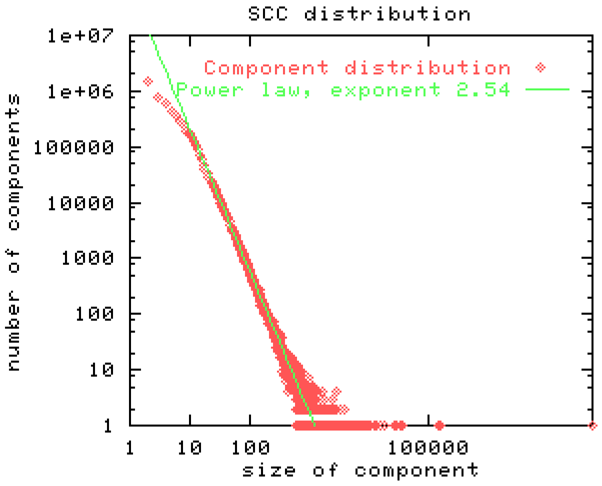
The bow-tie structure of the Web

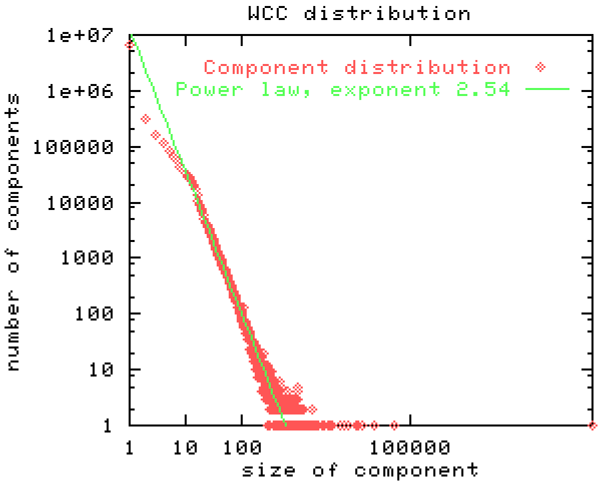




SCC and WCC distribution

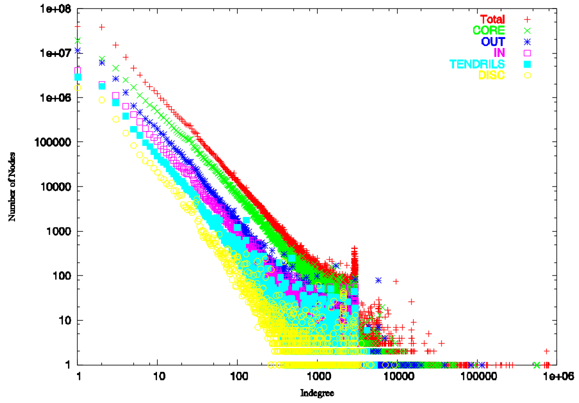
- The SCC and WCC sizes follows a power law distribution
  - the second largest SCC is significantly smaller

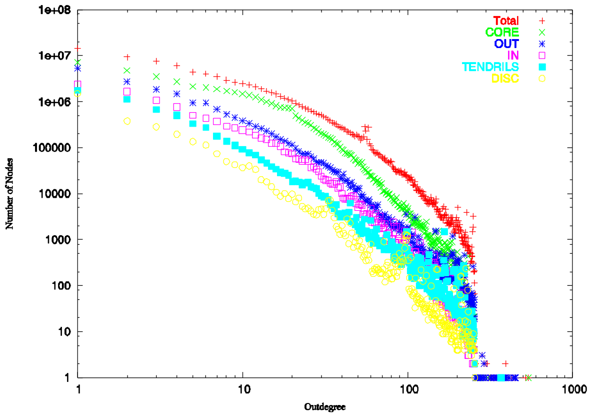


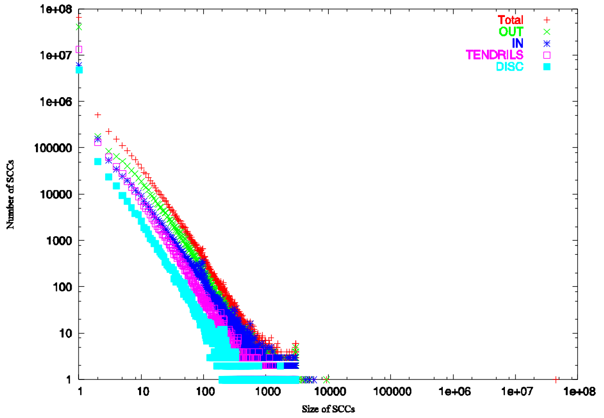


The inner structure of the bow-tie

- What do the individual components of the bow tie look like?
  - They obey the same power laws in the degree distributions

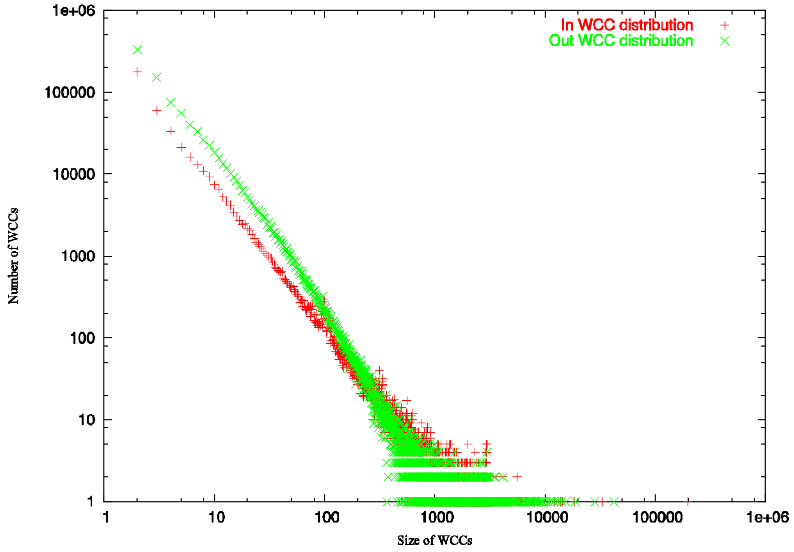






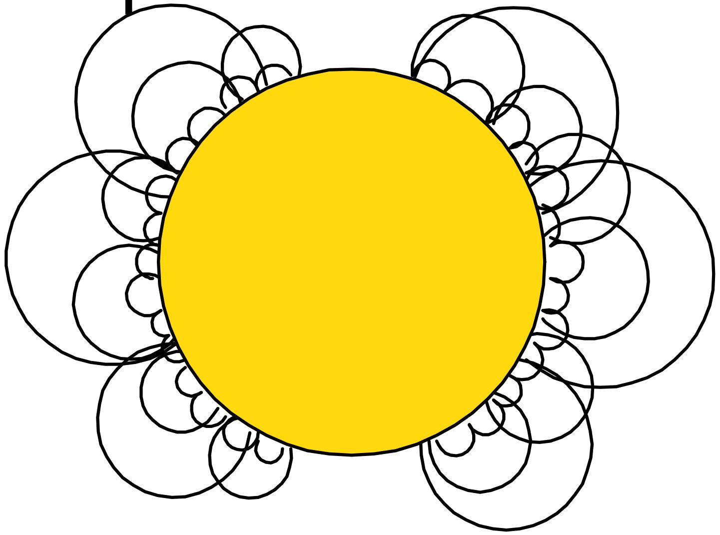
The inner structure of the bow-tie

- Is it the case that the bow-tie repeats itself in each of the components (self-similarity)?
  - It would look nice, but this does not seem to be the case
  - no large WCC, many small ones



The daisy structure?

- Large connected core, and highly fragmented IN and OUT components



- Unfortunately, we do not have a large crawl to verify this hypothesis

A different kind of self-similarity [Dill et al]

- Consider Thematically Unified Clusters (TUC): pages grouped by
  - keyword searches
  - web location (intranets)
  - geography
  - hostgraph
  - random collections

- All such TUCs exhibit a bow-tie structure!

Self-similarity

- The Web consists of a collection of self-similar structures that form a backbone of the SCC

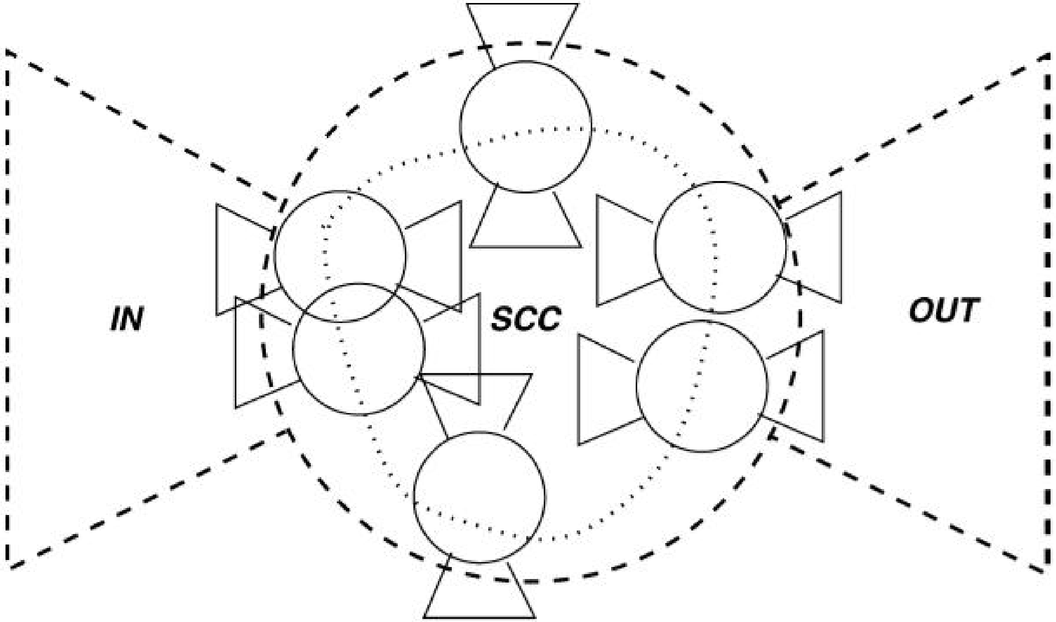
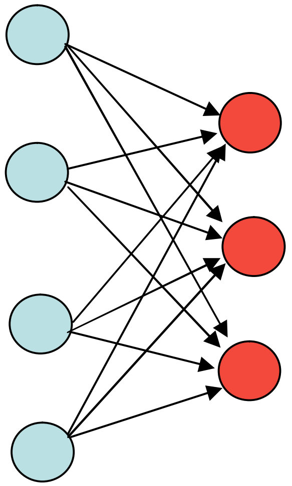


Fig. 4. TUCs connected by the navigational backbone inside the SCC of the Web graph.

Community discovery [Kumar et al]

- Hubs and authorities
  - hubs: pages that point to (many good) pages
  - authorities: pages that are pointed to by (many good) pages

- Find the (i,j) bipartite cliques of hubs and authorities
  - intuition: these are the core of a community
  - grow the core to obtain the community

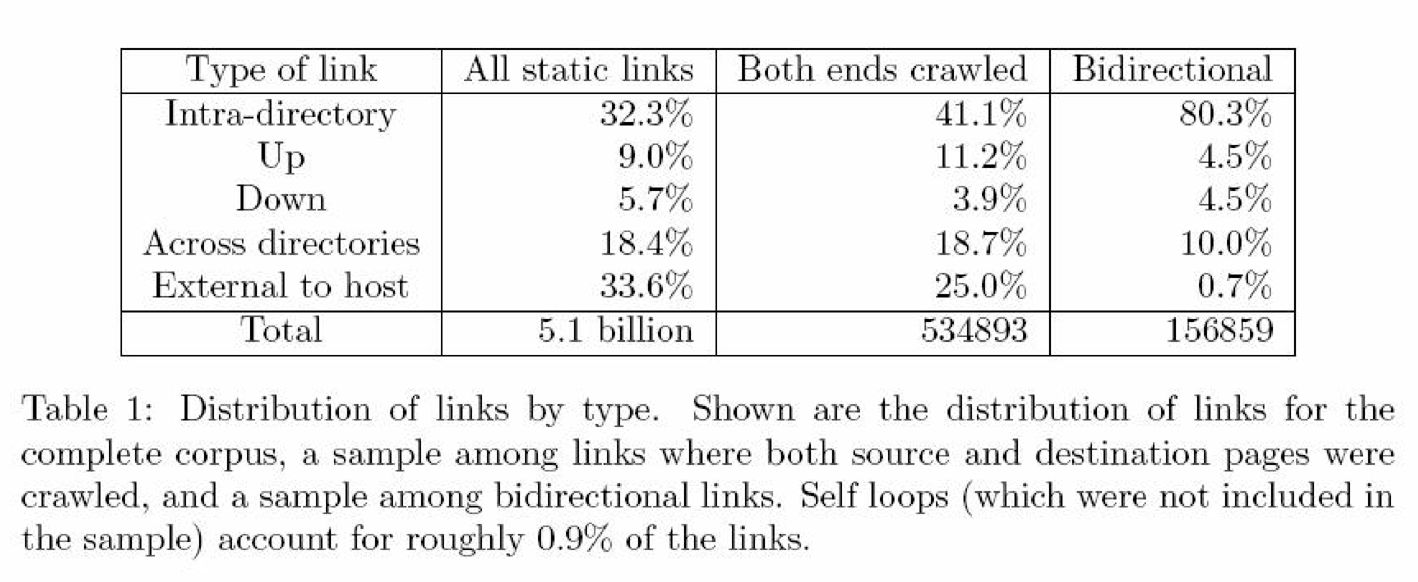


Bipartite cores

- Computation of bipartite cores requires heuristics for handling the Web graph
  - iterative pruning steps
- Surprisingly large number of bipartite cores
  - lead to the copying model for the Web
- Discovery of unusual communities of enthusiasts
  - Australian fire brigadiers

Hierarchical structure of the Web [Eiron and McCurley]

- The links follow in large part the hierarchical structure of the file directories
  - locality of links



Type of link	All static links	Both ends crawled	Bidirectional
Intra-directory	32.3%	41.1%	80.3%
Up	9.0%	11.2%	4.5%
Down	5.7%	3.9%	4.5%
Across directories	18.4%	18.7%	10.0%
External to host	33.6%	25.0%	0.7%
Total	5.1 billion	534893	156859

Table 1: Distribution of links by type. Shown are the distribution of links for the complete corpus, a sample among links where both source and destination pages were crawled, and a sample among bidirectional links. Self loops (which were not included in the sample) account for roughly 0.9% of the links.

Web graph representation

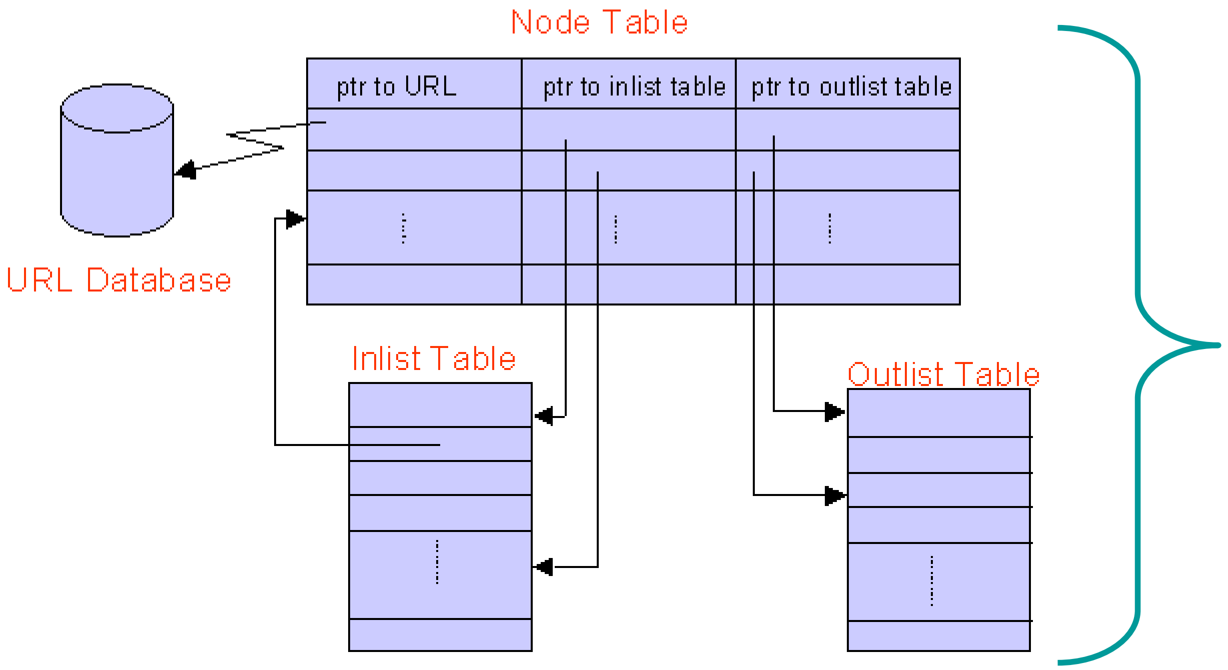
- How can we store the web graph?
  - we want to compress the representation of the web graph and still be able to do random and sequential accesses efficiently.
  - for many applications we need also to store the transpose

Links files

- A sequence a records
- Each record consists of a source URL followed by a sequence of destination URLs

```
http://www.foo.com/
    http://www.foo.com/css/foostyle.css
    http://www.foo.com/images/logo.gif
    http://www.foo.com/images/navigation.gif
    http://www.foo.com/about/
    http://www.foo.com/products/
    http://www.foo.com/jobs/
```

A simple representation



also referred to as starts table, or offset table

the link database

The URL Database

- Three kinds of representations for URLs
  - Text: original URL
  - Fingerprint: a 64-bit hash of URL text
  - URL-id: sequentially assigned 32-bit integer

URL-ids

- Sequentially assigned from 1 to N
- Divide the URLs into three partitions based on their degree
  - indegree or outdegree > 254, high-degree
  - 24 - 254, medium degree
  - Both <24, low degree
- Assign URL-ids by partition
- Inside each partition, by lexicographic order

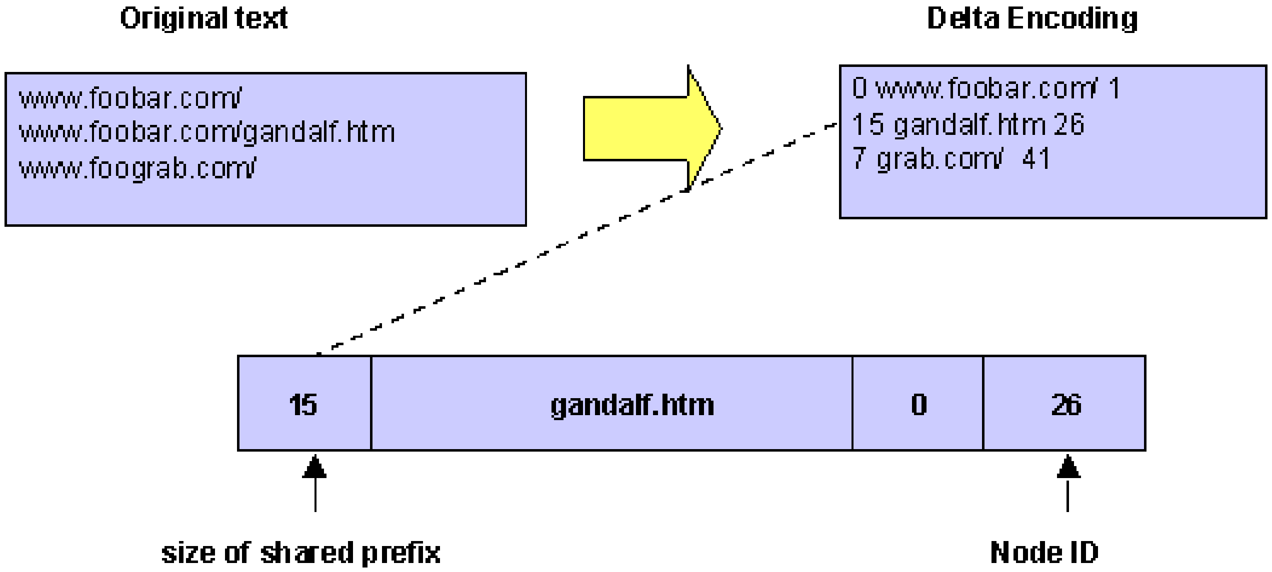
Compression of the URL database [BBHKV98]

- When the URLs are sorted lexicographically we can exploit the fact that consecutive URLs are similar

  www.foobar.com
  www.foobar.com/gandalf

- delta-encoding: store only the differences between consecutive URLs

delta-encoding of URLS



Original text

www.foobar.com/
www.foobar.com/gandalf.htm
www.foograb.com/

Delta Encoding

0 www.foobar.com/ 1
15 gandalf.htm 26
7 grab.com/ 41

15	**gandalf.htm**	**0**	**26**

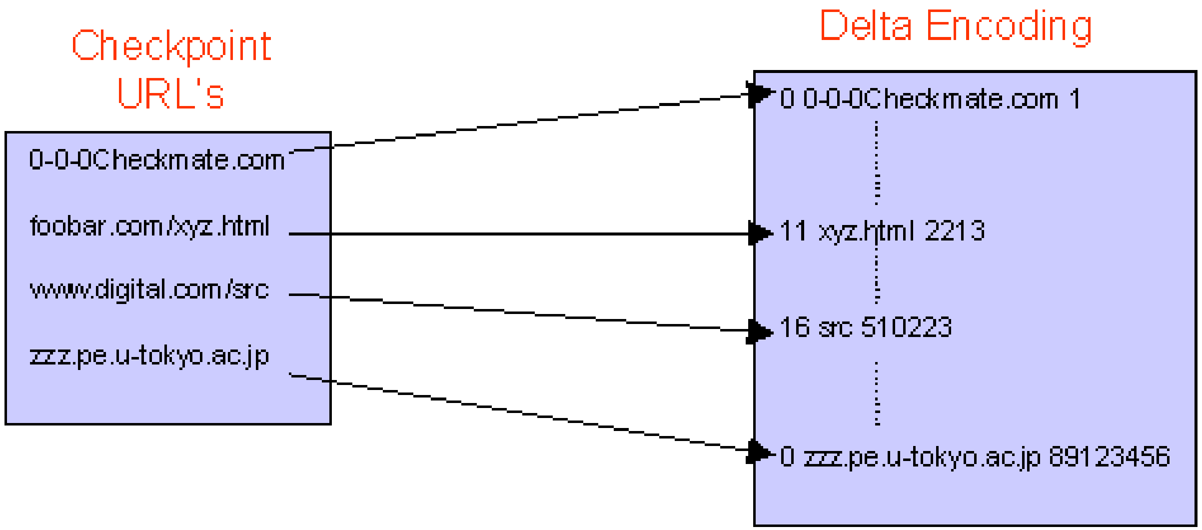
size of shared prefix

Node ID

- problem: we may have to traverse long reference chains

Checkpoint URLs

- Store a set of Checkpoint URLs
  - we first find the closest Checkpoint URL and then go down the list until we find the URL



Checkpoint URL's

0-0-0Checkmate.com

foobar.com/xyz.html

www.digital.com/src

zzz.pe.u-tokyo.ac.jp

Delta Encoding

0 0-0-0Checkmate.com 1

11 xyz.html 2213

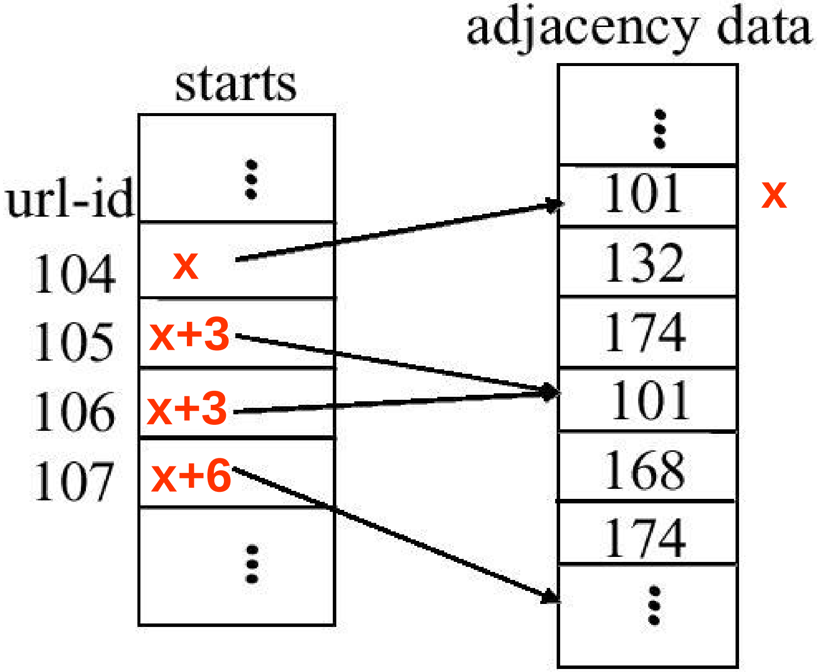
16 src 510223

0 zzz.pe.u-tokyo.ac.jp 89123456

  - results in 70% reduction of the URL space

The Link Database [RSWW]

- Maps from each URL-id to the sets of URL-ids that are its out-links (and its in-links)



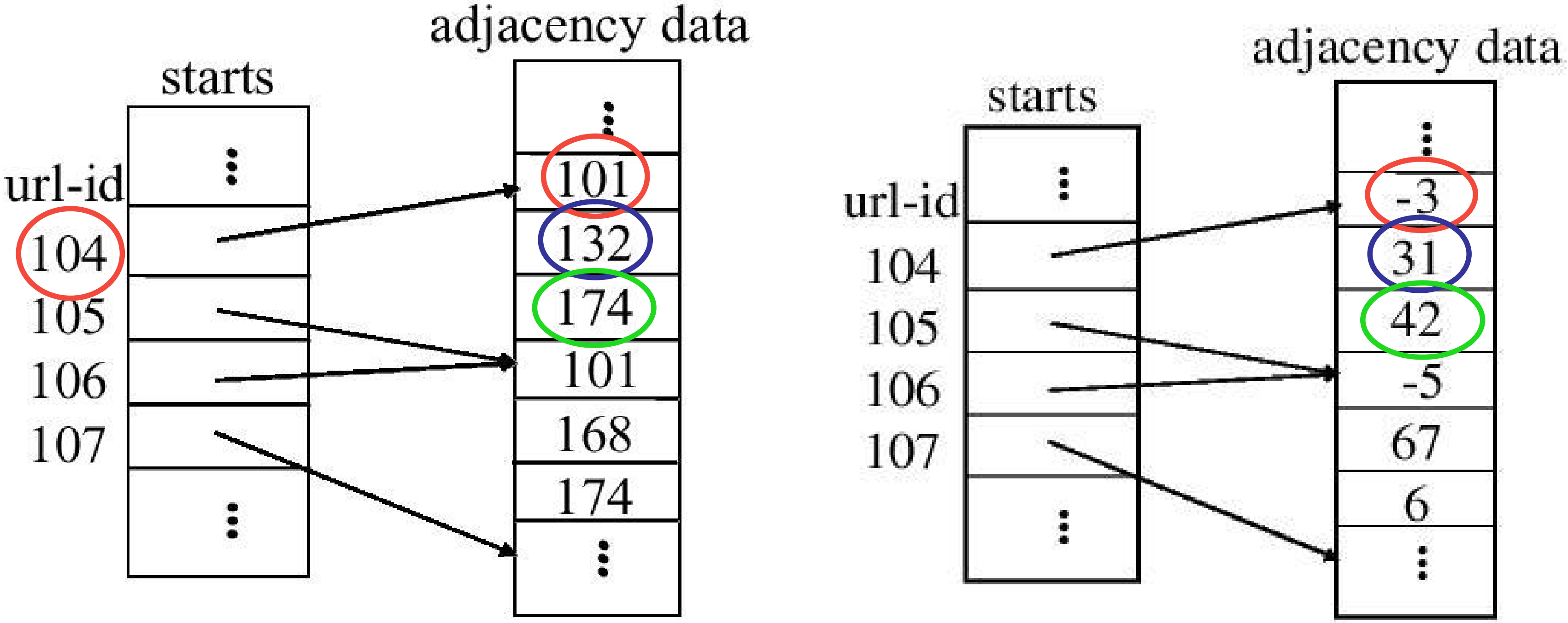
Vanilla representation

- Avg 34 bits per in-link
- Avg 24 bits per out-link

Compression of the link database

- We will make use of the following properties
  - Locality: usually most of the hyperlinks are local, i.e, they point to other URLs on the same host. The literature reports that on average 80% of the hyperlinks are local.
  - Lexicographic proximity: links within same page are likely to be lexicographically close.
  - Similarity: pages on the same host tend to have similar links (results in lexicographic proximity on the in-links)
- How can we use these properties?

delta-encoding of the link lists



-3 = 101 - 104

31 = 132 - 101

42 = 174 - 132

How do we represent deltas?

- Any encoding is possible (e.g. Huffman codes) – it affects the decoding time.
- Use of Nybbles
  - nybble: four bits, last bit is 1 if there is another nybble afterwards. The remaining bits encode an unsigned number

28 = 0111 1000

  - if there are negative numbers then the least significant bit (of the useful bits) encodes the sign

28 = 1111 0000

-6 = 0011 1010

-28 = 1111 0010

Compressing the starts array

- For the medium and small degree partitions, break the starts array into blocks. In each block the starts are stored as offsets of the first index
    - only 8 bits for the small degree partition, 16 bits for the medium degree partition
    - considerable savings since most nodes (about 74%) have low degree (power-law distribution)

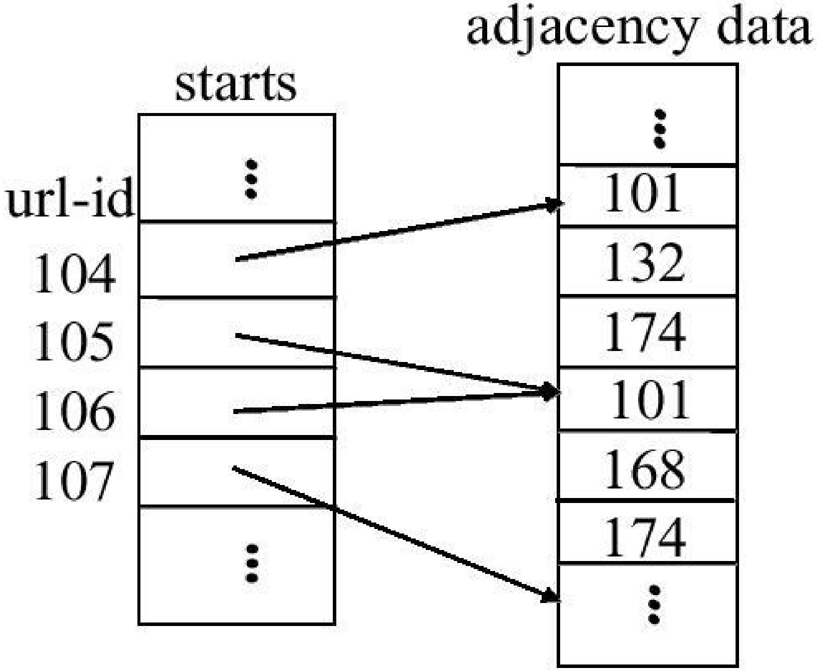
- Intuition: for low and med partitions the starts will be close to each other

Resulting compression

- Avg 8.9 bits per out-link
- Avg 11.03 bits per in-link

We can do better

- Any ideas?



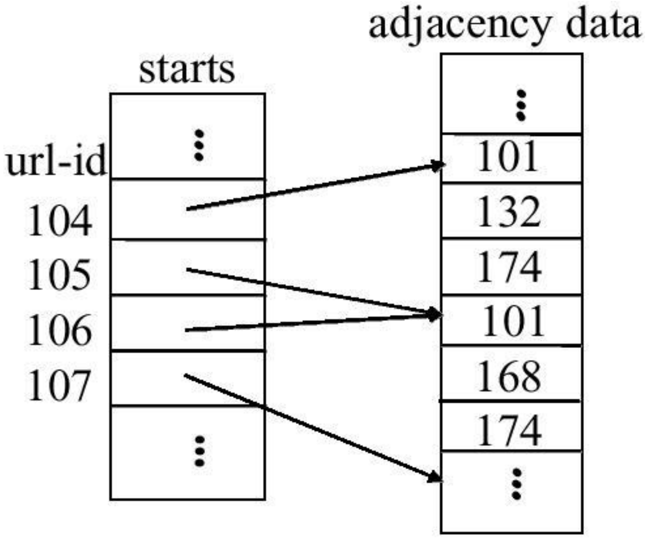
101, 132, 174

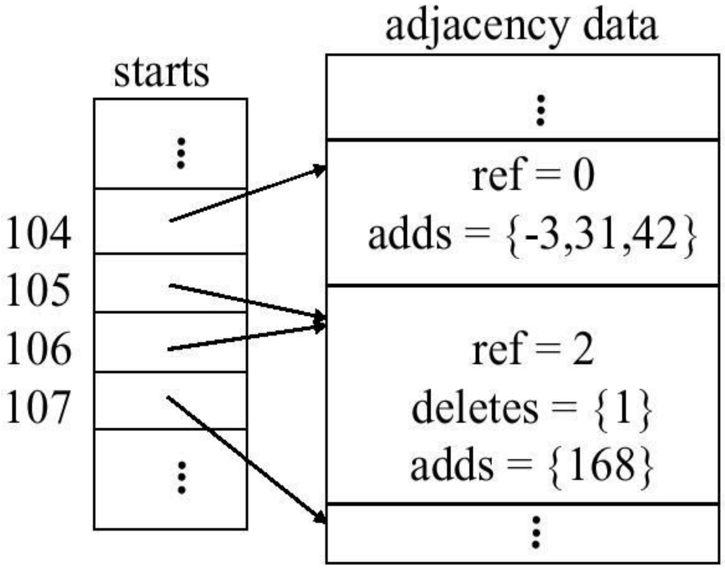
101, 168, 174

Reference lists

- Select one of the adjacency lists as a reference list
- The other lists can be represented by the differences with the reference list
  - deleted nodes
  - added nodes

Reference lists





Interlist distances

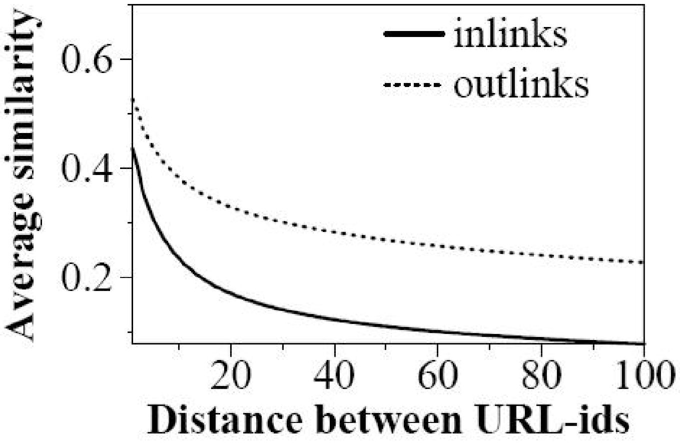
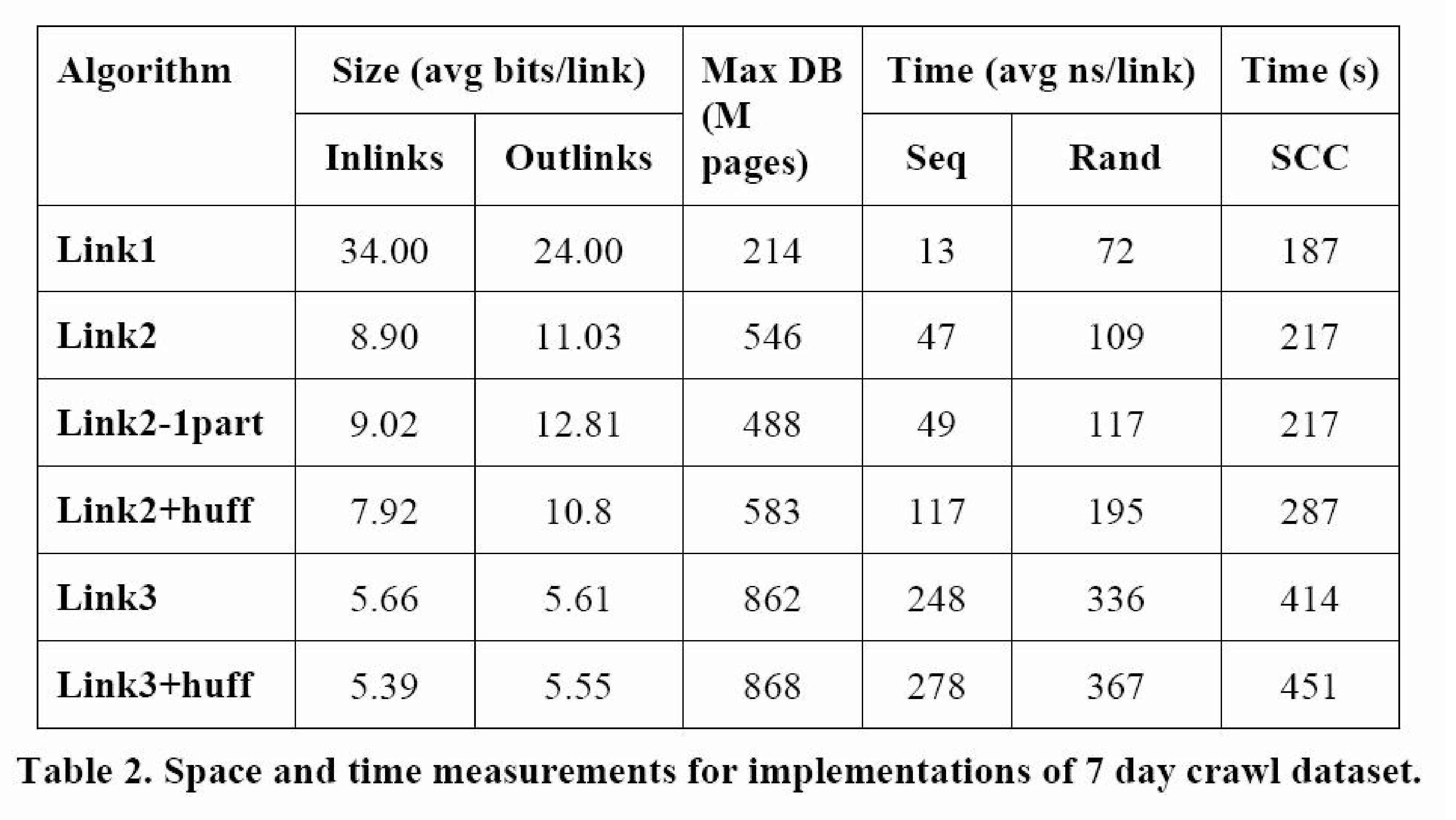


Figure 4: Similarity between neighboring adjacency lists

- Pages that with close URL-ids have similar lists
- Resulting compression
  - Avg 5.66 bits per in-link
  - Avg 5.61 bits per out-link

Space-time tradeoffs



Algorithm	Size (avg bits/link)		Max DB (M pages)	Time (avg ns/link)		Time (s)
	Inlinks	Outlinks		Seq	Rand	SCC
Link1	34.00	24.00	214	13	72	187
Link2	8.90	11.03	546	47	109	217
Link2-1part	9.02	12.81	488	49	117	217
Link2+huff	7.92	10.8	583	117	195	287
Link3	5.66	5.61	862	248	336	414
Link3+huff	5.39	5.55	868	278	367	451

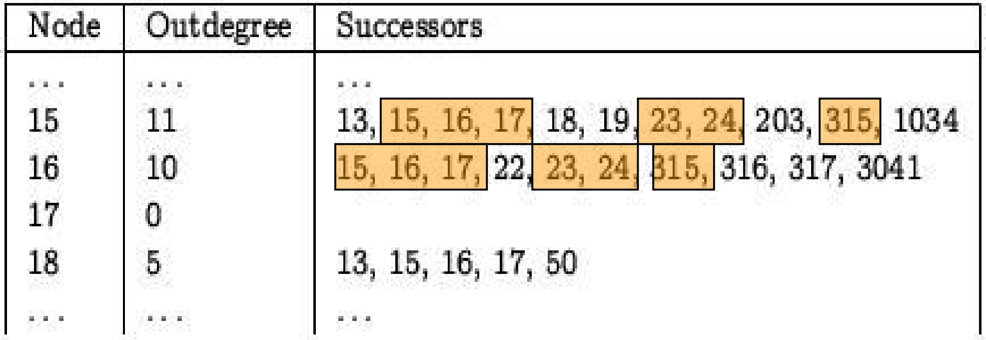
Table 2. Space and time measurements for implementations of 7 day crawl dataset.

Exploiting consecutive blocks [BV04]

- Many sets of links correspond to consecutive blocks of URL-ids. These can be encoded more efficiently

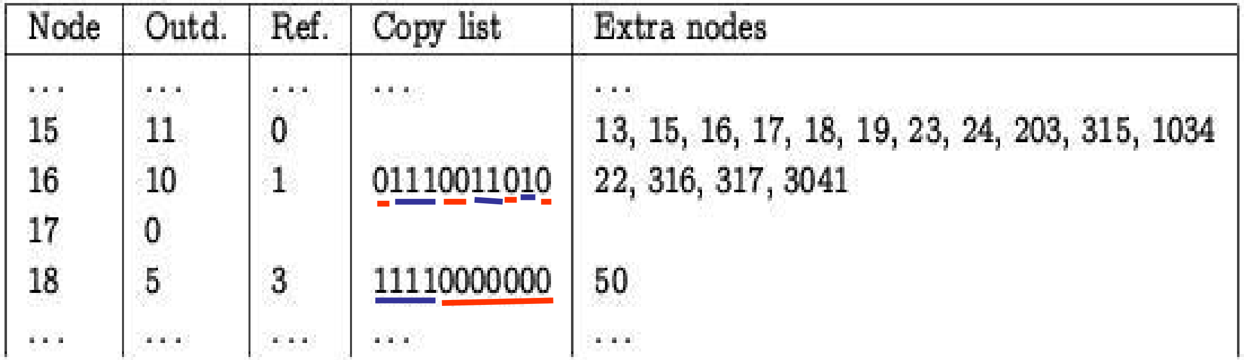
Interlist compression

Uncompressed link list



Node	Outdegree	Successors
...	...	...
15	11	13, 15, 16, 17, 18, 19, 23, 24, 203, 315, 1034
16	10	15, 16, 17, 22, 23, 24, 315, 316, 317, 3041
17	0	
18	5	13, 15, 16, 17, 50
...	...	...

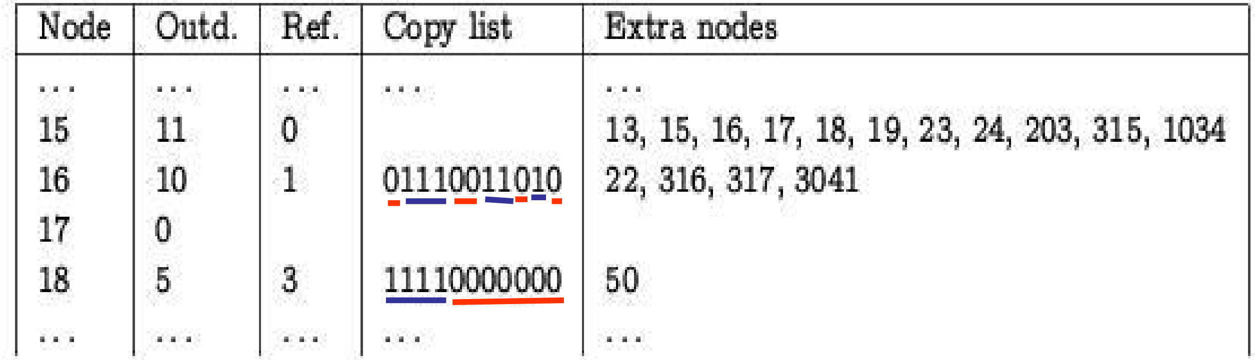
Interlist compression



Node	Outd.	Ref.	Copy list	Extra nodes
...	...	...	...	...
15	11	0		13, 15, 16, 17, 18, 19, 23, 24, 203, 315, 1034
16	10	1	01110011010	22, 316, 317, 3041
17	0			
18	5	3	11110000000	50
...	...	...	...	...

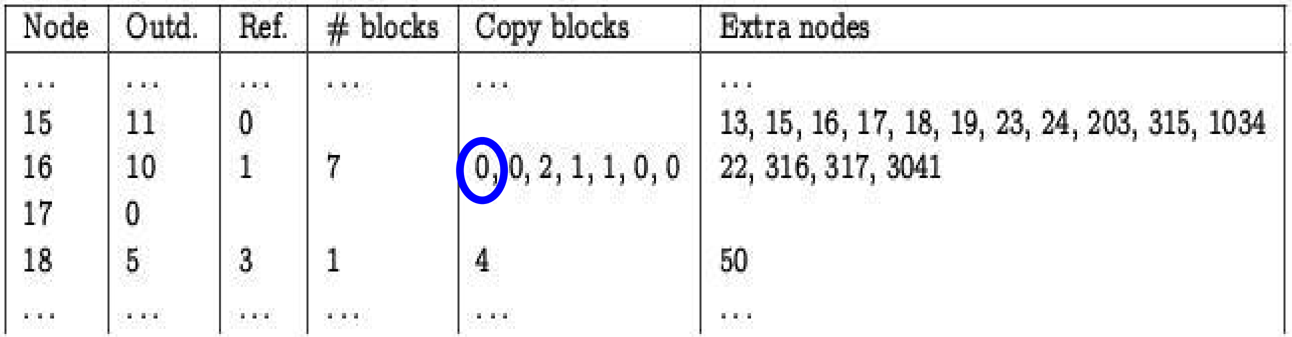
Compressing copy blocks

Interlist compression



Node	Outd.	Ref.	Copy list	Extra nodes
...	...	...	...	...
15	11	0		13, 15, 16, 17, 18, 19, 23, 24, 203, 315, 1034
16	10	1	01110011010	22, 316, 317, 3041
17	0			
18	5	3	11110000000	50
...	...	...	...	...

Adjacency list with copy blocks.



Node	Outd.	Ref.	# blocks	Copy blocks	Extra nodes
...	...	...	...	...	...
15	11	0			13, 15, 16, 17, 18, 19, 23, 24, 203, 315, 1034
16	10	1	7	0, 0, 2, 1, 1, 0, 0	22, 316, 317, 3041
17	0				
18	5	3	1	4	50
...	...	...	...	...	...

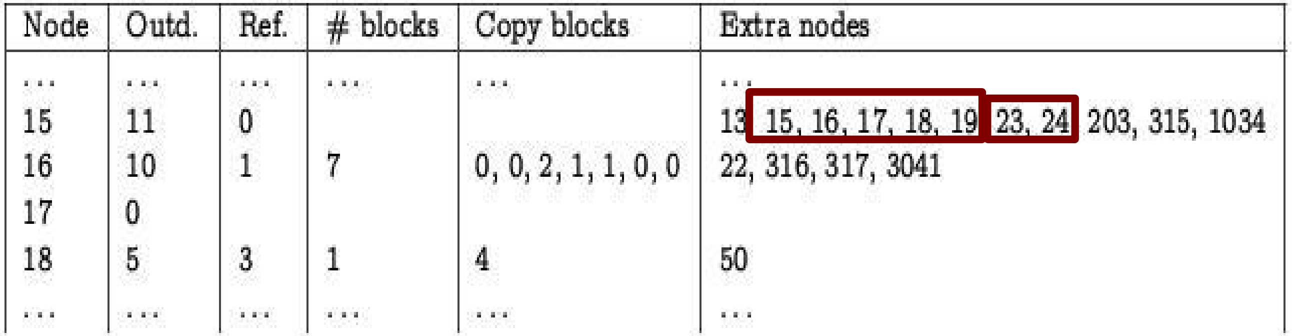
The last block is omitted;

The first copy block is 0 if the copy list starts with 0;

The length is decremented by one for all blocks except the first one.

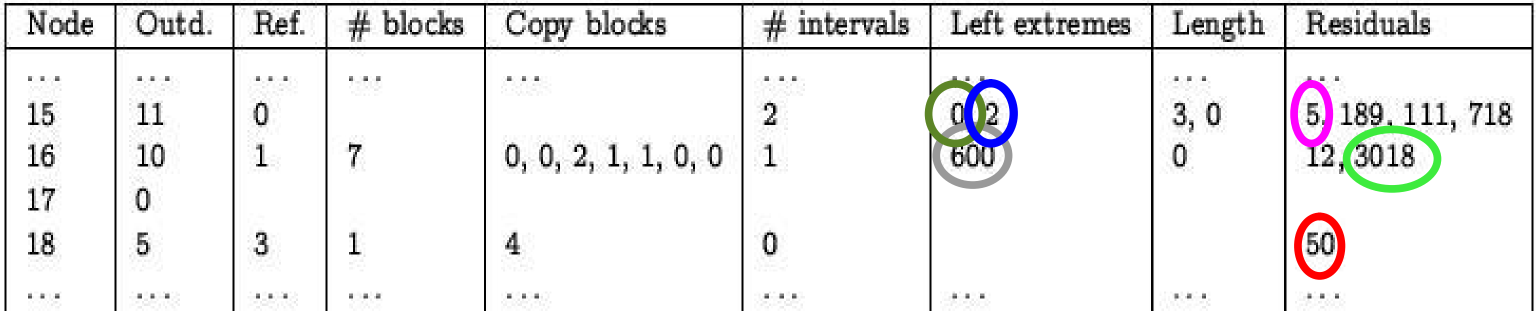
Compressing intervals

Adjacency list with copy blocks.



Node	Outd.	Ref.	# blocks	Copy blocks	Extra nodes
…	…	…	…	…	…
15	11	0			13, 15, 16, 17, 18, 19, 23, 24, 203, 315, 1034
16	10	1	7	0, 0, 2, 1, 1, 0, 0	22, 316, 317, 3041
17	0				
18	5	3	1	4	50
…	…	…	…	…	…

Adjacency list with intervals.



Node	Outd.	Ref.	# blocks	Copy blocks	# intervals	Left extremes	Length	Residuals
…	…	…	…	…	…	…	…	…
15	11	0			2	0, 2	3, 0	5, 189, 111, 718
16	10	1	7	0, 0, 2, 1, 1, 0, 0	1	600	0	12, 3018
17	0							
18	5	3	1	4	0			50
…	…	…	…	…	…	…	…	…

Intervals: represented by their left extreme and length;
Intervals length: are decremented by the threshold Lmin (=2);
Residuals: compressed using differences.

$$\nu(x) = \begin{cases} 2x & \text{if } x \geq 0 \\ 2|x| - 1 & \text{if } x < 0 \end{cases}$$

for the first residual value

0 = (15-15)*2

600 = (316-16)*2

5 = |13-15|*2-1

3018 = 3041-22-1

2 = 23 -19 -2

Resulting compression

- Avg 3.08 bits per in-link
- Avg 2.89 bits per out-link

Acknowledgements

- Thanks to Adrei Broder, Luciana Buriol, Debora Donato, Stefano Leonardi for slides material

References

- K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public Web search engines. Proc. 7th International World Wide Web Conference, 1998.
- M. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. On Near-Uniform URL Sampling . 9th International World Wide Web Conference, May 2000.
- S. Lawrence, C. L. Gilles, Searching the World Wide Web, Science 280, 98-100 (1998).
- A. Albert, H. Jeong, and A.-L. Barab -bási, Diameter of the World Wide Web, Nature,401, 130-131 (1999).
- A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener. Graph structure in the web. 9th International World Wide Web Conference, May 2000.
- S. Dill, R. Kumar, K. McCurley, S. Rajagopalan, D. Sivakumar, A. Tomkins. Self-similarity in the Web. 27th International Conference on Very Large Data Bases, 2001.
- R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for cyber communities, Proc. 8th WWW , Apr 1999.
- Nadav Eiron and Kevin S. McCurley, Locality, Hierarchy, and Bidirectionality on the Web, Workshop on Web Algorithms and Models, 2003.
- D. Donato, S. Leonardi, P. Tsaparas, Mining the inner structure of the Web, WebDB 2005.
- A. Gulli and A. Signorini. The indexable web is more than 11.5 billion pages. In Proceedings of 14th International World Wide Web Conference, Chiba, Japan, 2005.
- [RSWW] K. Randall, R. Stata, R. Wickremesinghe, J. Wiener, The Link Database: Fast Access to Graphs of the Web, Technical Report
- [BBHKV98] K. Bharat, A. Broder, M. Henzinger, P. Kumar, and S. Venkatasubramanian. *The connectivity server: fast access to linkage information on the web,* Proc. 7th WWW, 1998.
- [BV04] P. Boldi, S. Vigna, The Webgraph framework I: Compression Techniques, WWW 2004