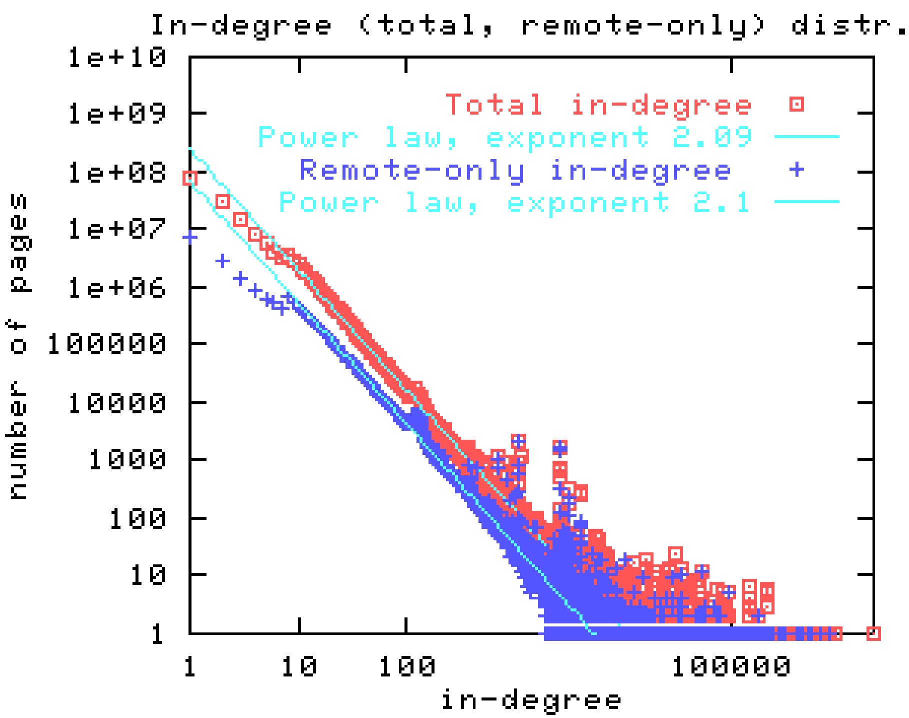
The Web is a Graph

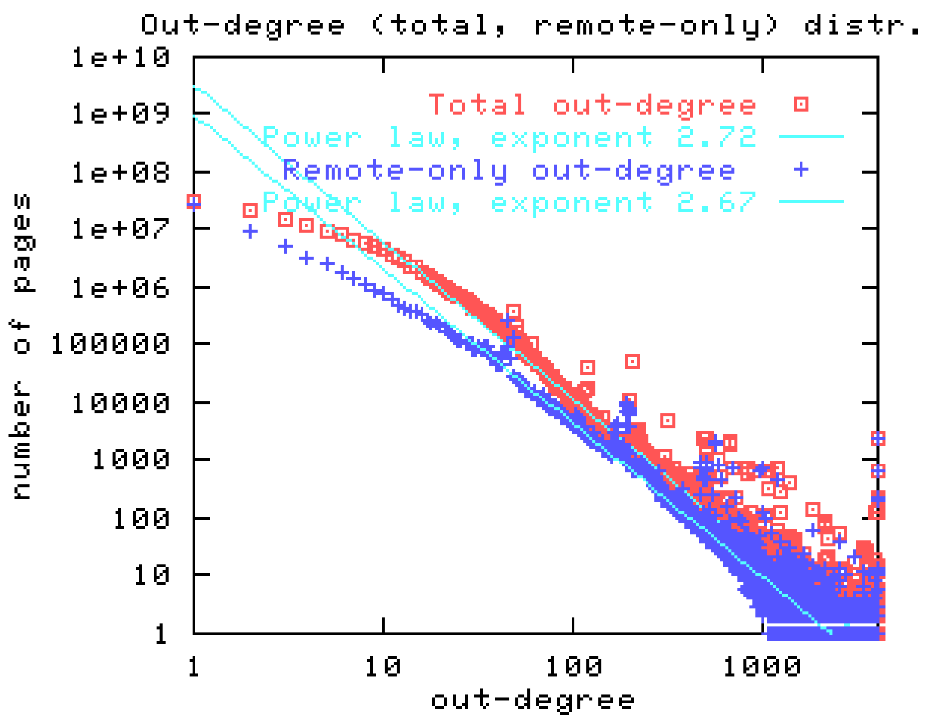
- pages are nodes, hyperlinks are edges
- Interesting Questions:
  - What is the distribution of in- and out-degrees?
  - How is its connectivity structure?
  - What is the diameter of the Web?
- Connectivity server (Bharat et al. 98)
  - Inverted index enriched with efficient data structures for hyperlink information (in-links and out-links)
- Detailed analysis of graph structure (Broder et al. 00)
  - Using an Altavista crawl (May 1999) with 203 million URLs and 1466 million links (all of which fit in 9.5 GB of storage)
  - Breadth-first search that reaches 100M nodes took about 4 minutes (on an improved version of the Connectivity Server)

In-Degree and Out-Degree

- Power law of in(out) degree:

  the probability that a node has in(out)-degree i is proportional to $1/i^x$ for some $x > 1$.

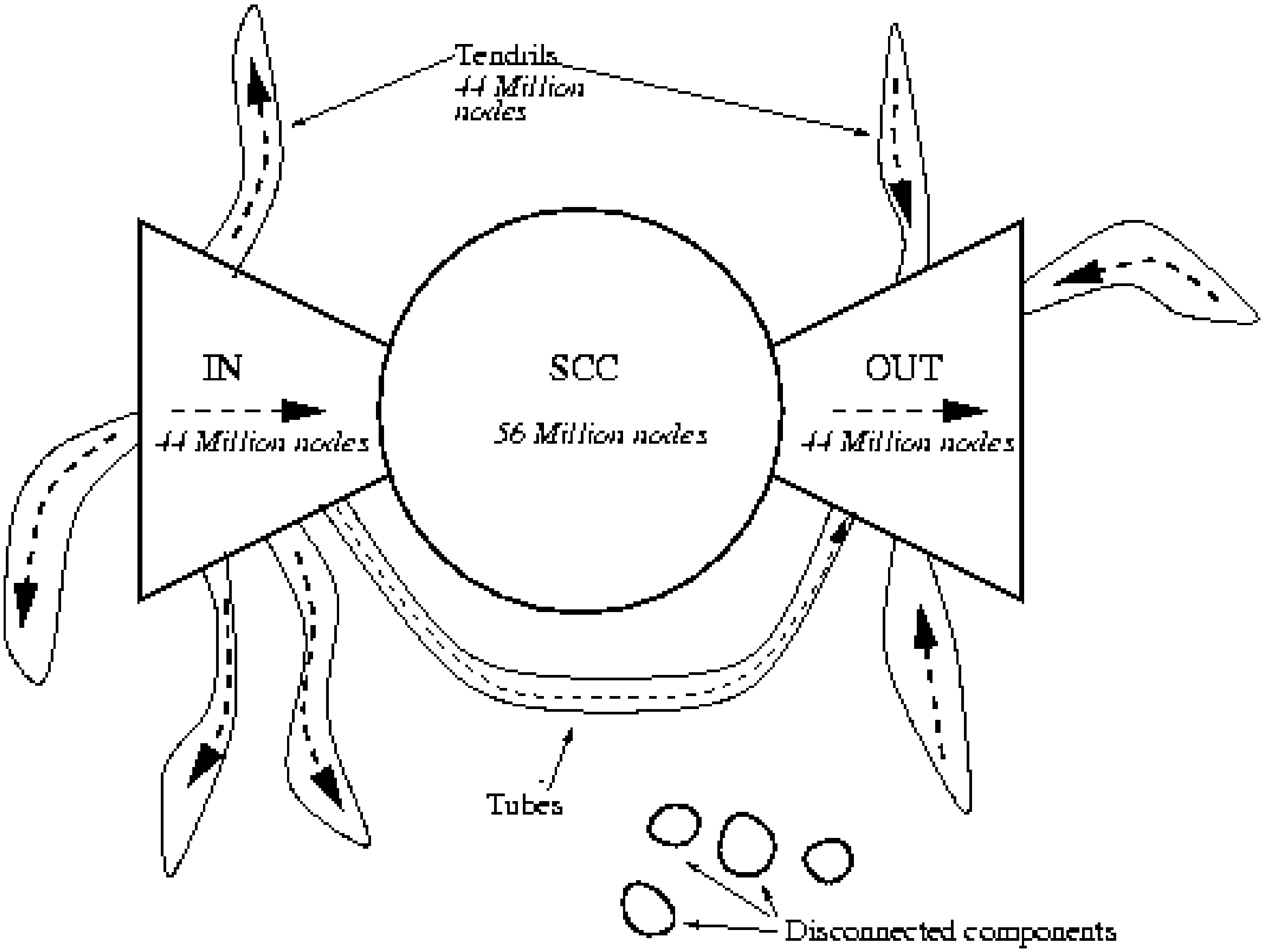




Connectivity

- Weakly connected components:
  - links are considered to be undirected
  - about 90% form a single component
- Strongly connected components:
  - only directed links
  - about 28% form a strongly connected core set of pages
  - number of strongly connected components also follows power law
- Diameter:
  - diameter of strongly connected core is > 27
  - diameter of the entire graph is > 500
  - probability that a path between two randomly selected pages exists is 0.24

Structure of the Web

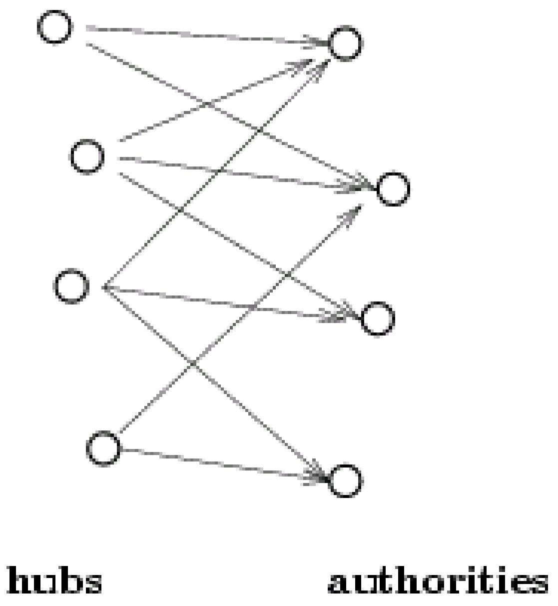


Finding relevant pages

- Search engines:
  - consult inverted index
  - return pages that match some or all query terms
- Problem:
  - query results are often too large to be inspected by user
- Need:
  - sorting according to relevance
- Limitations of Text-based approaches:
  - query terms may occur on non-relevant pages as well (maybe more frequently or more prominently)
  - query terms may not occur on a relevant page
  - queries as "short documents" do not provide good similarity scores
  - November 1997: (Brin & Page)
    only one of four top search engines finds itself!

Hubs & Authorities

- Authorities:
  - Pages that contain a lot of information about the query topic
- Hubs:
  - Pages that contain a large number of links to pages that contain information about the topic
- Mutual reinforcement:
  - A good *hub* points to many good *authorities*
  - A good *authority* is pointed to by many good *hubs*



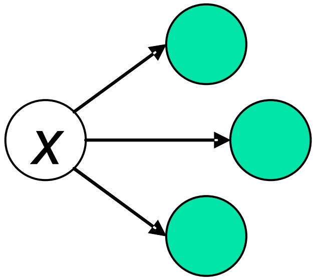
Using Graph Structure to Determine Relevance

- simple approach:
  - sort query results according to number of in-links
  - *Problem:* universally popular pages would be considered to be highly authorative for all search terms they contain
- HITS: Algorithm for identifying good hub and authority pages for a query
  - each page is associated with a *hub score* and an *authority score*
  - scores are computed based on graph structure of the Web
  - mutual reinforcement of hubs and authorities is exploited with an *iterative algorithm*

Hub and Authority Scores

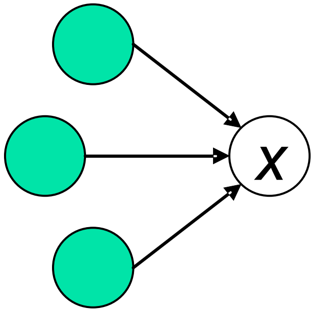
- Hub Scores h(p):
  - hub scores are updated with the sum of all authority weights of pages it points to

$$h(x) = \sum_{(x,y) \in E} a(y)$$



- Authority Scores a(p):
  - authority scores are updated with the sum of all hub weights that point to it

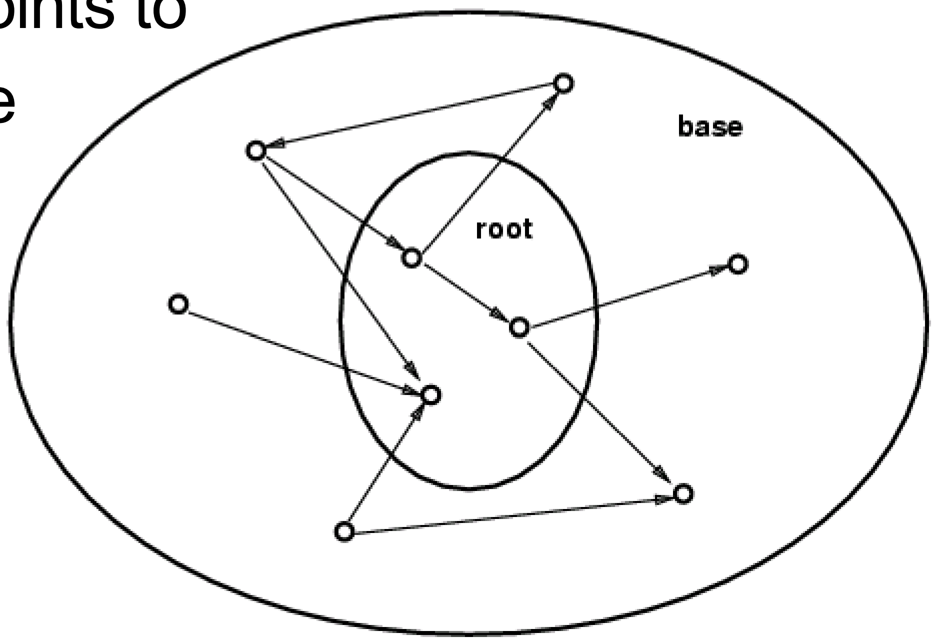
$$a(x) = \sum_{(y,x) \in E} h(y)$$



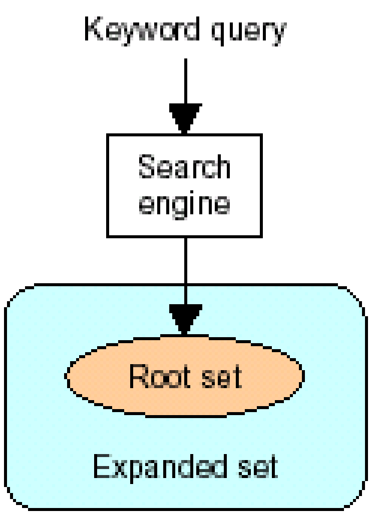
- Iterative Computation:
  - normalize weights
  - repeat update
  - convergence can be proven

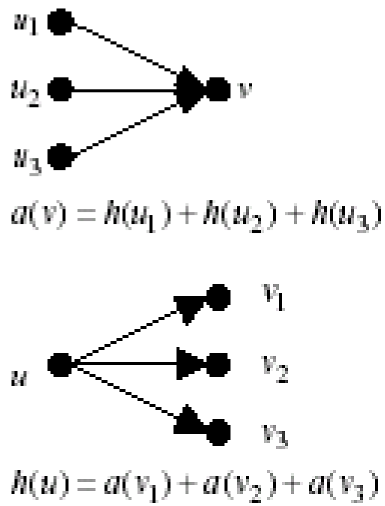
HITS algorithm (Kleinberg, 1997)

- collect the *root set*
  - first t hits from a conventional search engine (typically $t = 200$)
- construct a *base set*
  - include all pages the root set points to
  - include pages that point into the root set ($< d$ for each page in the root set, typically $d = 50$)
  - size ~ 1000 - 5000
- construct a *focused subgraph*
  - graph structure of the base set
  - delete *intrinsic* links (i.e., links between pages in same domain)
- iteratively compute hub and authority scores



The HITS algorithm





$\vec{a} \leftarrow (1, \ldots, 1)^T, \vec{h} \leftarrow (1, \ldots, 1)^T$
while $\vec{h}$ and $\vec{a}$ change 'significantly' do
  $\vec{h} \leftarrow E\vec{a}$
  $\ell_h \leftarrow \|\vec{h}\|_1$
  $\vec{h} \leftarrow \vec{h}/\ell_h$
  $\vec{a} \leftarrow E^T\vec{h}_0 = E^T E\vec{a}_0$
  $\ell_a \leftarrow \|\vec{a}\|_1$
  $\vec{a} \leftarrow \vec{a}/\ell_a$
end while

$||h||$ and $||a||$ are L_1 vector norms
E is the neighborhood matrix
a converges to the principal eigenvector of E^TE
h converges to the principal eigenvector of EE^T

Problems

- Efficiency
  - construction of graph has to be performed on-line
- Irrelevant links
  - Advertisements
  - Automatically generated links
- Mutually reinforcing relationship between hosts
  - multiple documents on one site pointing to document D at another drives up their hub scores and the authority score of D
- Topic Drift
  - documents in base set may be too general (e.g. Jaguar -> car)

Improvements (Bharat & Henzinger 98)

- Improved Connectivity Analysis:
  - normalize score by number of links between different hosts
  - *authority weights*:
    - weight a link with $1/k$ if there are k documents from the same site pointing to the authority
  - *hub weights*:
    - weight a link with $1/k$ if the hub points to k documents on the same host
- Relevance Weights:
  - compute a pseudo-document of first 1000 words of each document in root set
  - only include documents in base set that have a minimum similarity to the pseudo-document
  - weight propagation is weighted by relevance weight

Page Rank (Brin & Page, 1998)

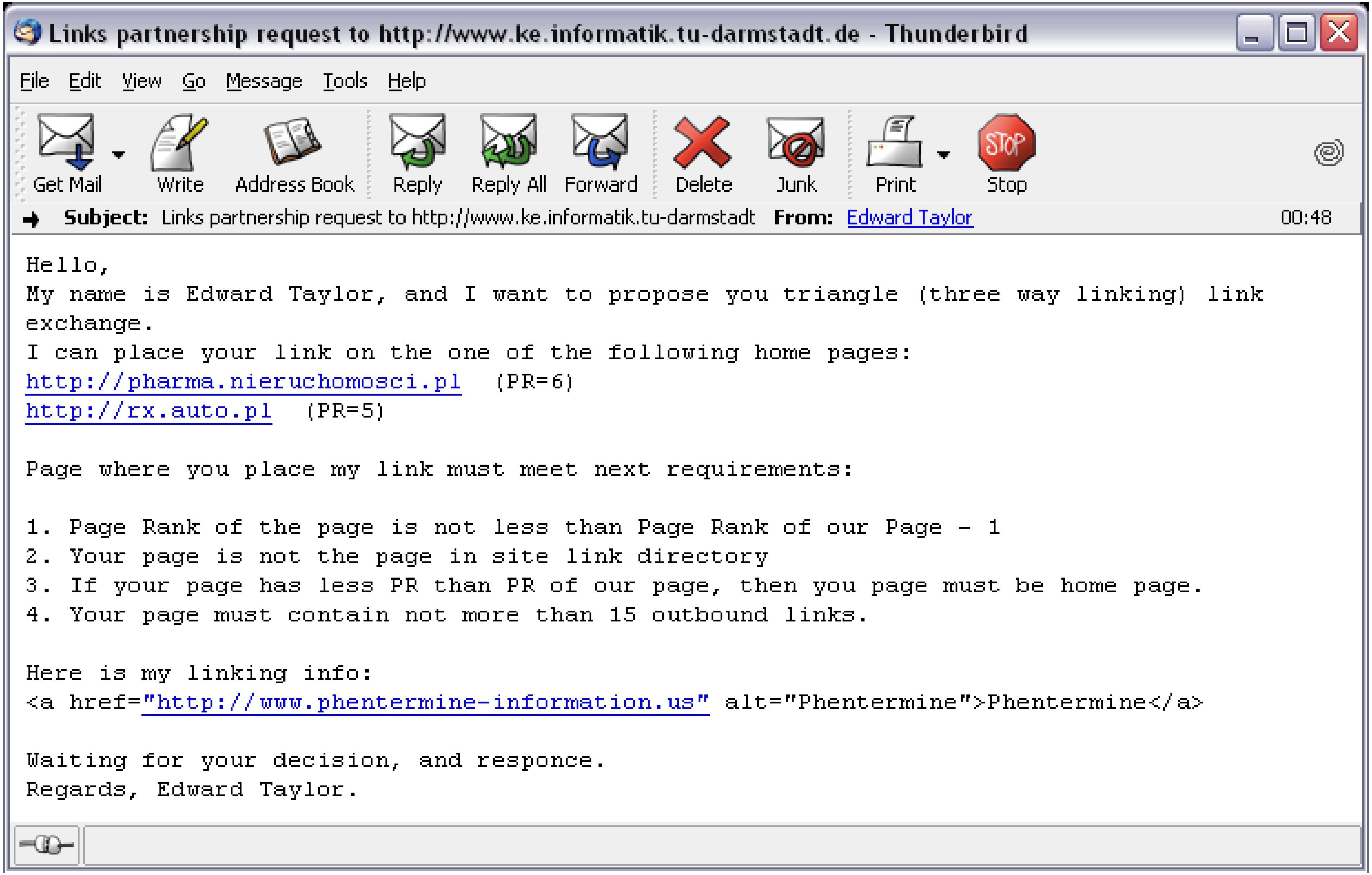
- Idea: model of a random surfer
  - clicks on one of the outgoing links at random
  - with a probability d jump to a random page on the Web
- PageRank $pr(p)$:

$$pr(p) = (1-d)\frac{1}{N} + d \sum_{(q,p)\in E} \frac{pr(q)}{o(q)}$$

$o(p)$	out degree of page p
d	damping factor (0.85)
N	total number of pages

- page rank prefers pages that have
  - a large in-degree
  - predecessors with a large page rank
  - predecessors with a small out-degree
- page rank is a probability distribution over pages

Link Spam



Hello,
My name is Edward Taylor, and I want to propose you triangle (three way linking) link exchange.
I can place your link on the one of the following home pages:
http://pharma.nieruchomosci.pl (PR=6)
http://rx.auto.pl (PR=5)

Page where you place my link must meet next requirements:

1. Page Rank of the page is not less than Page Rank of our Page - 1
2. Your page is not the page in site link directory
3. If your page has less PR than PR of our page, then you page must be home page.
4. Your page must contain not more than 15 outbound links.

Here is my linking info:
Phentermine

Waiting for your decision, and responce.
Regards, Edward Taylor.

Google (status ~ 1998)

- Design goal: High precision in relevance sorting
- Ranking is based on combination of several factors
  - PageRank weights
    - iterative PageRank computations
    - off-line, for 26 million pages in several hours
  - matches in anchor texts
  - proximity information
  - assigns different weights to different types of hits
    - font size, font face, URL, title, ...
- Tuning the weights for the combiner is a "black art"
  - earlier versions used feedback of "trusted" users

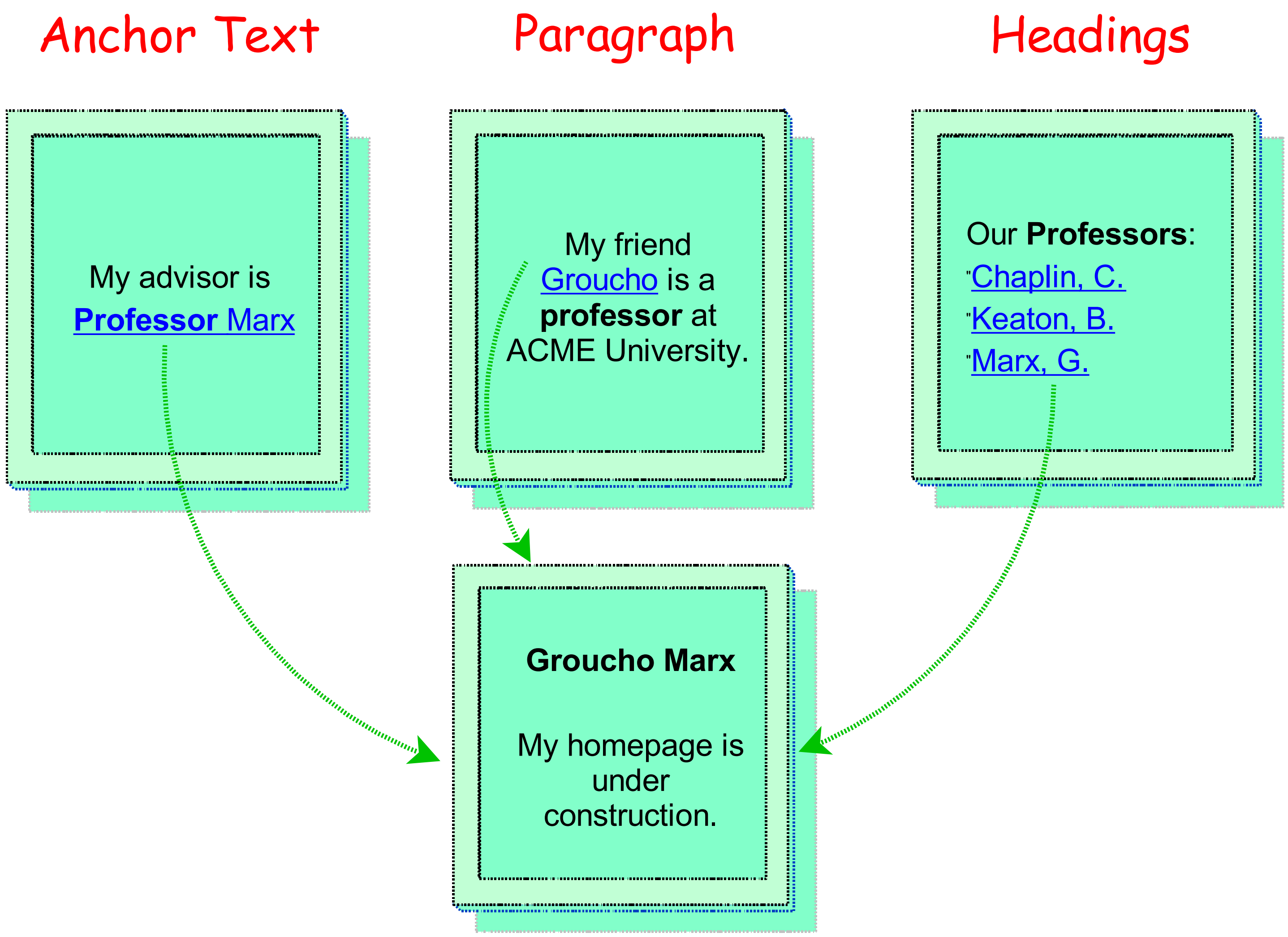
PageRank vs HITS

- PageRank advantage over HITS
  - Query-time cost is low
    - HITS: computes an eigenvector for every query
  - Less susceptible to localized link-spam
- HITS advantage over PageRank
  - HITS ranking is sensitive to query
  - HITS has notion of hubs and authorities
- Topic-sensitive PageRanking
  - Attempt to make PageRanking query sensitive
  - Basic idea: Tele-Portation (random jump) is topic-sensitive

Google Games

- Google Bombing
  - increasing a page's importance by adding links from different sites to it (e.g., in blogs)
  - possibly connected with spurious information
  - e.g., “miserable failure” / “völlige Inkompetenz”
- Google Whacking
  - try to find 2 English dictionary words that return a single hit
  - example: “masterfully incubatory” (http://www.googlewhack.com)
- Google Fight
  - try 2 keywords / phrases and see which one gets more hits
  - real applications: e.g., spelling correction
- BananaSlug
  - add random keywords to your query to get unexpected results

Hypertext Classification



Text vs. Links

- Text on WWW Pages may be
  - non-existent (images)
  - sparse
  - in an unknown language
  - misleading (false keywords)
  - irrelevant

- Links to WWW Pages provide
  - richer vocabulary (multiple authors)
  - redundancy
  - diversity through independent assessment of content
  - focus on important issues
  - multiple view points
  - multiple languages

Exploiting Hyperlink Structure

- Merging the Features:
  - join text of documents with (parts of) the text of the documents pointing to it
  - e.g., WWW Worm (McBryan 1994) indexes anchor text with the page it refers to
  - Chakrabarti et al. 1998 investigated this approach for hypertext classification (merging of full texts)
  - results got worse
- Use of Meta-Information: (Chakrabarti et al. 1998)
  - use *classification* of in-coming pages
  - iterative EM-like algorithm to converge to class assignments
  - produced somewhat better results
- Use of ILP (Craven & Slattery 1998, 2001)
  - represent Web graph in first-order logic
  - features of pages can be accessed via `link_to/2` relation

Labeling hypertext graphs: Scenario

- Snapshot of the Web graph, Graph $G = (V,E)$
- Set of topics,
- Small subset of nodes V_k labeled
- Use the supervision to label some or all nodes in $V - V_k$

21

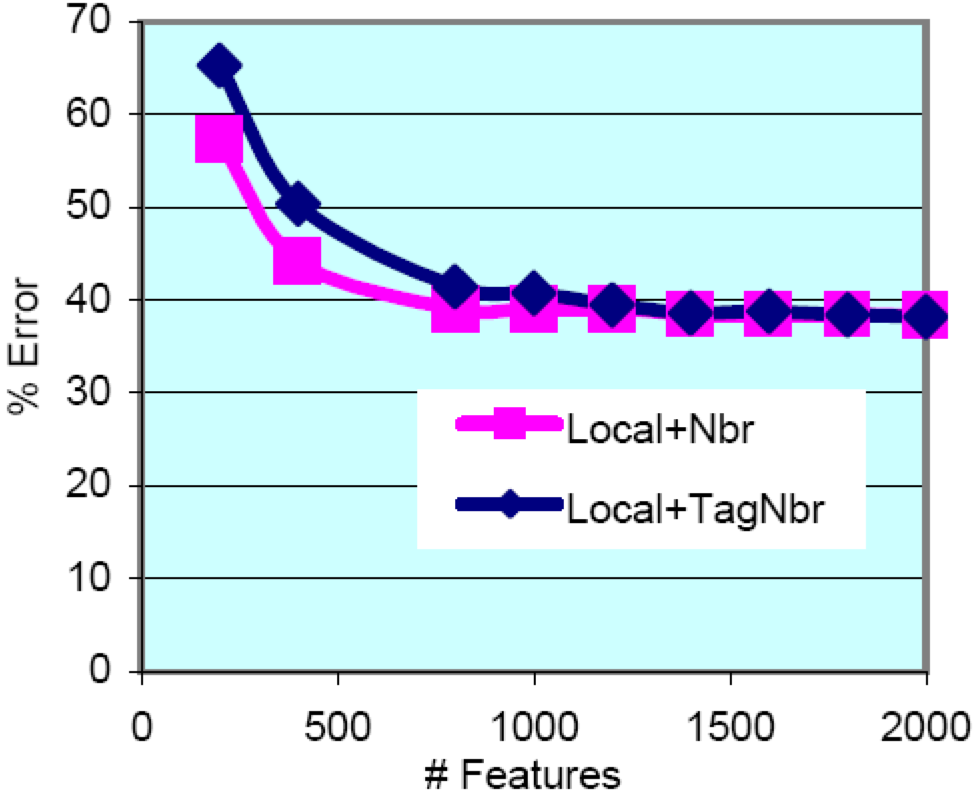
Absorbing features from neighboring pages

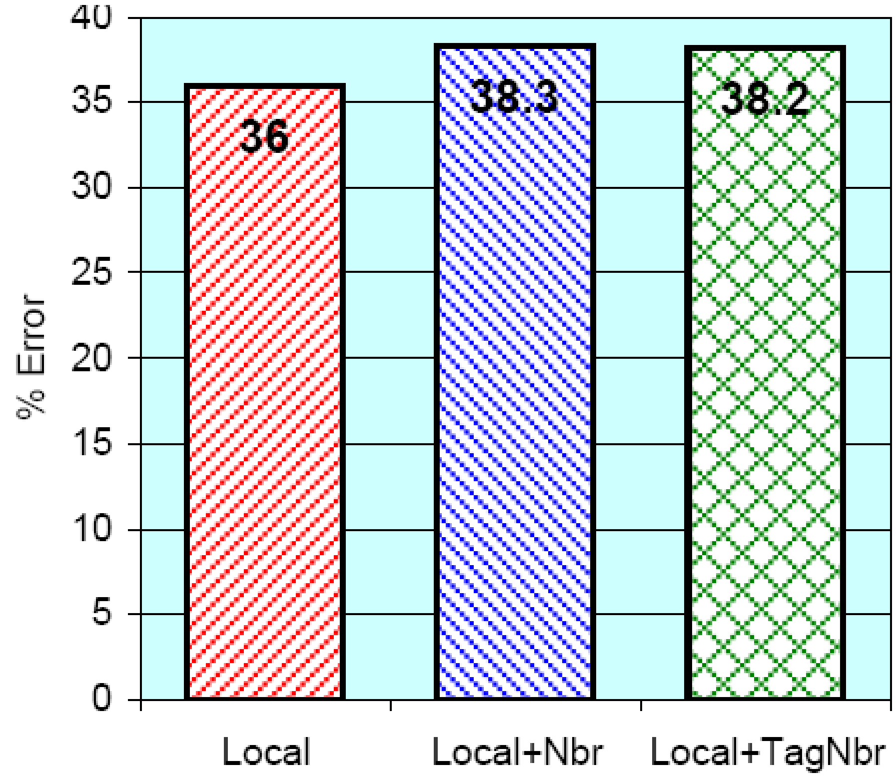
- A Page may have little text on it to train or apply a text classifier
- but it may reference other pages
- Often second-level pages have usable quantities of text
- Question: How to use these features ?

Absorbing features

- Indiscriminate absorption of neighborhood text does not help
  - At times even deteriorates accuracy
- Reason: Implicit assumption:
  - Topic of a page u is likely to be the same as the topic of a page cited by u.
  - Not always true
  - Topic may be “related” but not “same”
- Distribution of topics of the pages cited could be quite distorted compared to the totality of contents available from the page itself
- E.g.: university page with little textual content
  - Points to “how to get to our campus” or “recent sports prowess"

Absorbing text from neighboring pages



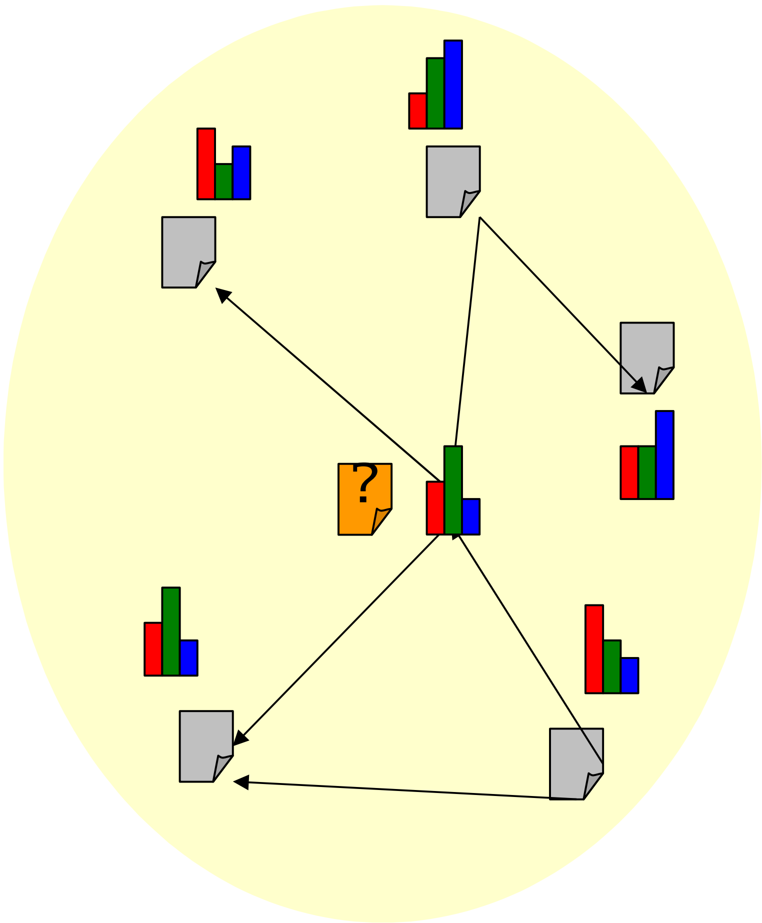


Absorbing text from neighboring pages in an indiscriminate manner does not help classify hyper-linked patent documents any better than a purely text-based naive Bayes classier.

Local: Only text of the page
Nbr: Merge text of page with text of all predecessor and successor pages
TagNbr: Maintain 3 separate sets of features: text of predecessors, local text, text of successors

Link-Derived Features

- *c*=class, *t*=text, *N*=neighbors
- Text-only model: Pr[*t*|*c*]
- Using neighbors' text to judge my topic: Pr[*t*, *t*(*N*) | *c*]
- Better model: Pr[*t*, *c*(*N*) | *c*]
  - use class distribution of N

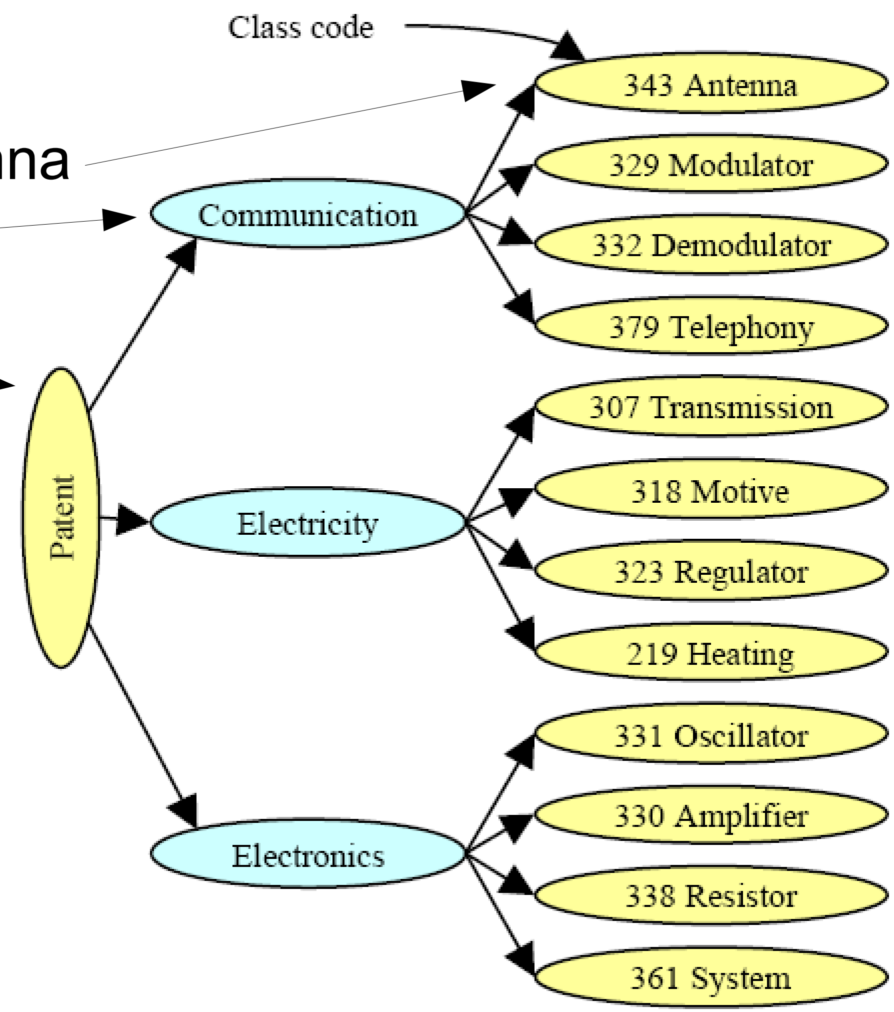


Absorbing link-derived features
(Chakrabarti, Dom, Indyk, 1998)

- Key insight 1
  - The classes of hyper-linked neighbors is a better representation of hyperlinks.
  - E.g.:
    - use the fact that u points to a page about athletics to raise our belief that u is a university homepage,
    - learn to systematically reduce the attention we pay to the fact that a page links to the Netscape download site.
- Key insight 2
  - class labels are from a is-a hierarchy.
    - evidence at the detailed topic level may be too noisy
    - coarsening the topic helps collect more reliable data on the dependence between the class of the homepage and the link-derived feature.

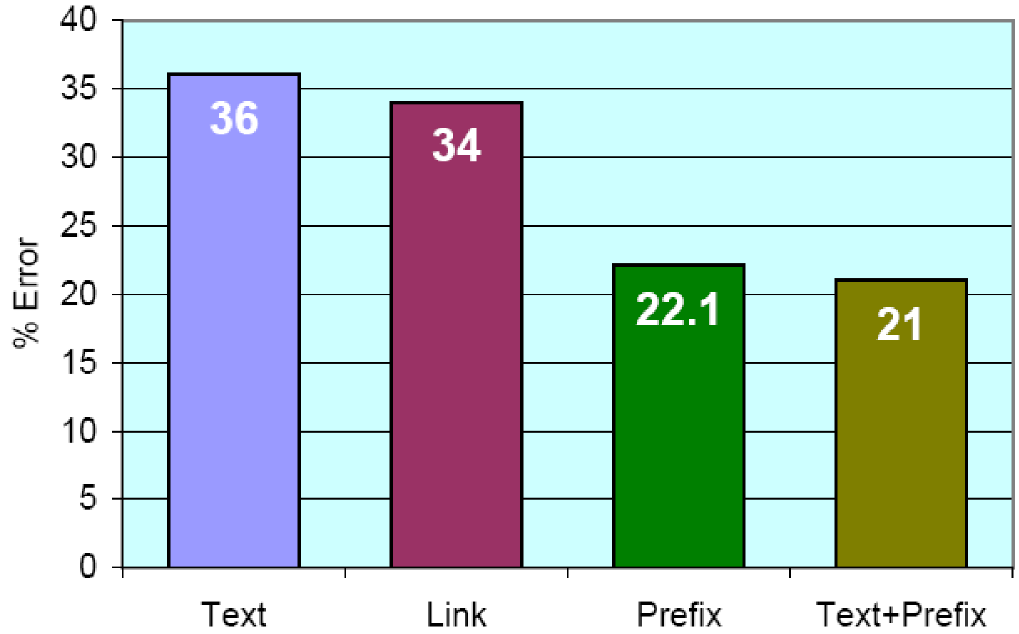
Absorbing link-derived features

- Add all prefixes of the class path to the feature pool:
  - Patent/Communication/343 Antenna
  - Patent/Communication
  - Patent
- Do feature selection to get rid of noise features
- Experiment
  - Corpus of US patents
  - Two level topic hierarchy
    - three first-level classes,
    - each has four children.
  - Each leaf topic has 800 documents,



Link-Derived Features: Results

- Experiment with
  - **Text** : only the Text on the page
  - **Link**: only all classes of neighboring pages
  - **Prefix:** classes of neighboring pages plus their prefixes
  - **Text+Prefix:** Text plus classes plus prefixes



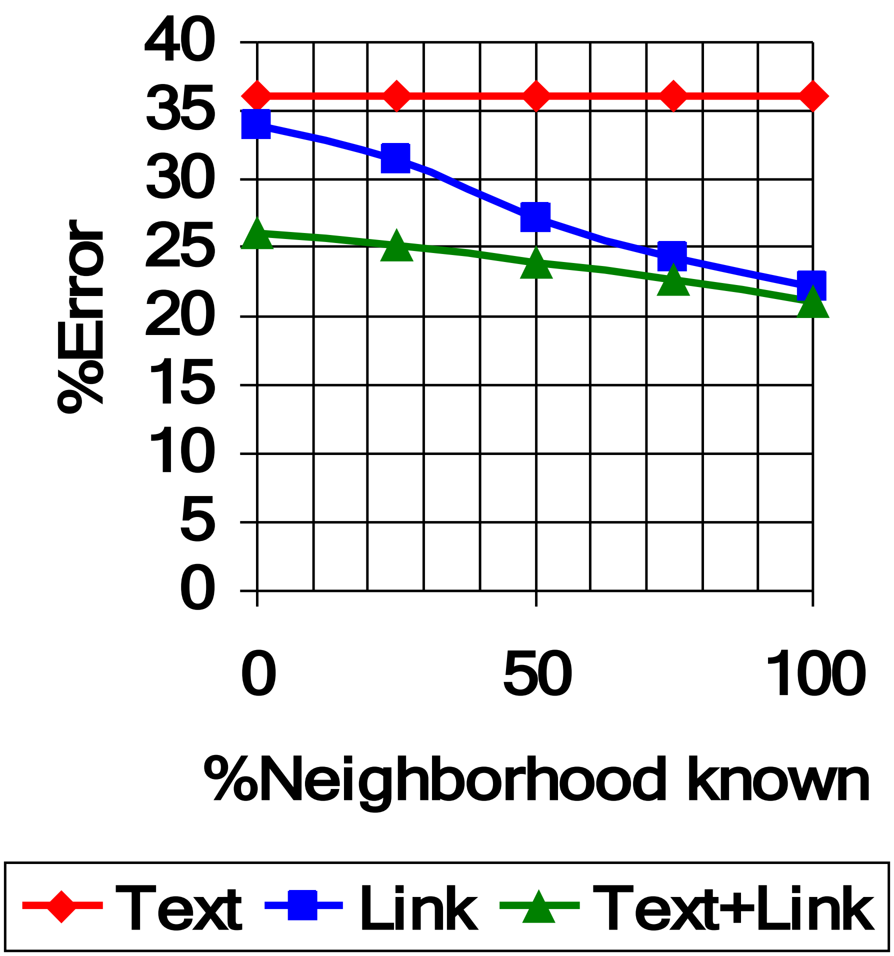
Using prefix-encoded link features in conjunction with text can significantly reduce classification error

Absorbing link-derived features: Limitation

- only a small subset is labeled ($|V^k| << |V|$)
- Hardly any neighbors of a node to be classified linked to any pre-labeled node
- Proposal
  - Start with a labeling of reasonable quality
    - Maybe using a text classifier
  - Do
    - Refine the labeling using a coupled distribution of text and labels of neighbors,
  - Until the labeling stabilizes.

Results

- 9600 patents from 12 classes marked by USPTO
- Patents have text and cite other patents
- Expand test patent to include neighborhood
- ‘Forget’ fraction of neighbors’ classes



Problems

- Features of predecessor pages should be kept separately
  - Chakrabarti's approach merges the entire text from all predecessor pages into a single pot
- Redundancy provided by multiple predecessors should be exploited
  - ILP approaches can (in principle) keep features separately, but focus on single discriminators
- Not the entire text of a predecessor page is relevant
  - each page is predecessor of several pages, in the worst case each belongs to a potentially different class -> each case should be represented differently
- Not all pages have relevant meta-information

Hyperlink Ensembles

I. Discard page text
II. Represent each link to a page as a separate example
   - use only part of the text (otherwise all links of the same page have identical representations, but may point to different targets)
III. Encode as Set-Valued Features:
   - **ANCHOR**: All words between `<A HREF...>` and `</A>`
   - **HEADING**: All words occurring in Headings that structurally precede the link
   - **PARAGRAPH**: All words of the paragraph that contains the link
IV. Ensemble formation:
   - one training example for each hyperlink
   - one ensemble of predictions for each page (one prediction for each of its predecessors)
   - combine predictions for each predecessor to a single prediction for the target page

Comparison to Full-Text Classifier

- Setup:
  - Ripper as base learner
  - WebKB, 1050 pages, 5803 links, 7 classes
- Results
  - full text uses about 20,000 features
  - the link classifier uses about 8,000 features
  - feature suset selection (using information gain) helps to improve the performance
  - link-based classifier are better anyways

Links **(Weight,All)**	82,67
Links **(Weight, A&H)**	85.14
Full Text	70.67
Text **(50% features)**	73.90
Text **(10% features)**	74.19
Text **(5% features)**	74,76
Text **(1% features)**	71,33
Text **(0.1% features)**	54.67

Feature Sets / Voting Schemes

- anchor text and headings are more important than text in paragraph around the link
- use of confidences is important for combining

	Vote	Weight	Max
Default	51.81	51.81	51.81
Anchor	67.52	74.19	74.76
Headings	60.48	72.95	72.95
Paragraph	63.05	66.95	66.29
Anchor & Hdgs.	74.48	85.14	86.57
Anchor & Par.	68.00	73.90	74.67
Headings & Par.	70.48	81.14	81.33
All	74.19	82.67	83.24

Gain through Ensemble

- comparison between accuracy on *predicting links* without (left) and with (right) combining predictions
- redundancy is exploited
- pages with more incoming links are classified more reliably

	Links	Weight
Default	36.67	36.67
Anchor	57.92	75.37
Headings	43.34	70.77
Paragraph	53.40	66.33
Anchor & Hdgs.	62.49	86.25
Anchor & Par.	58.40	73.46
Headings & Par.	58.50	80.30
All	57.99	79.44

Hyperlink Ensembles: Results

- using link and HTML structure can outperform text classifiers
  - anchor text and section headings are good complimentary features
  - weighting is important for combining predictors
  - successful exploitation of the redundancy provided by multiple links to a page
- some open questions:
  - How is the performance on other hypertext classification tasks?
  - Can ensemble techniques of this type be used for solving (certain types of) multi-instance problems?