Hidden-Web Databases: Classification and Search

Luis Gravano
Columbia University

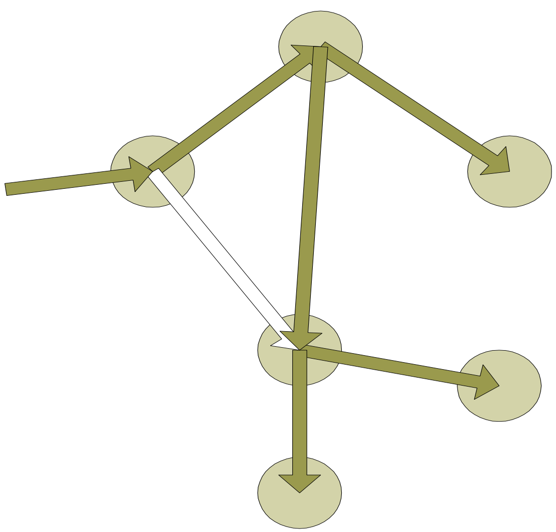
http://www.cs.columbia.edu/~gravano

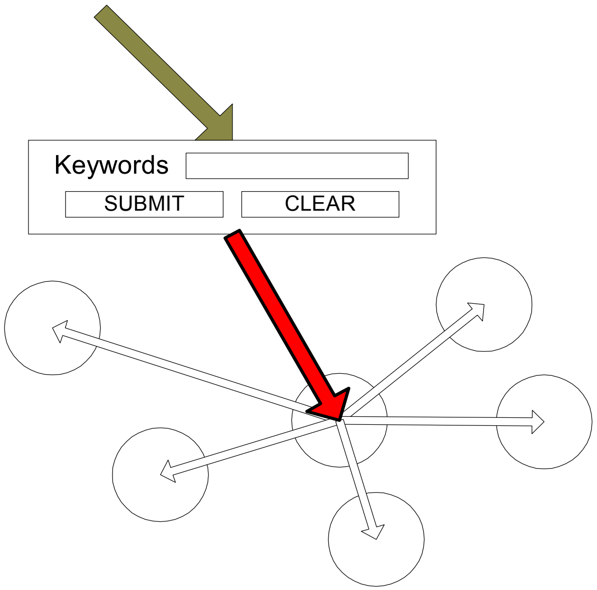
Joint work with **Panos Ipeirotis** (Columbia) and **Mehran Sahami** (Stanford/Google)

Outline of Talk

- **Classification of Hidden-Web Databases**
- Search over Hidden-Web Databases
- Overview of Columbia's Database Group

“Hidden Web” Databases





“Surface” Web

- Link structure
- Crawlable
- Documents indexed by search engines

“Hidden” Web

- No link structure
- Documents “hidden” in databases
- Documents not indexed by search engines
- Need to query each collection individually

PubMed/Medline: Example of a Hidden-Web Database

Query **[thrombopenia]** on **PubMed**: **24,826 hits.**

PubMed is at www.ncbi.nlm.nih.gov/PubMed/

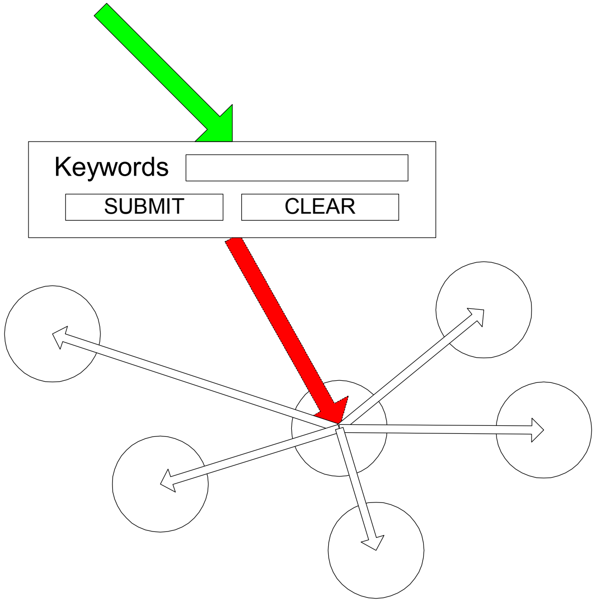
Query **[thrombopenia]** on **Google**: **856 hits.**

Query **[thrombopenia site:www.ncbi.nlm.nih.gov]** on **Google**: **0 hits.**

- Search engines ignore hidden-web databases (cannot crawl inside).
- Autonomous databases typically export no metadata.

Focus: Searchable Text Databases ("Hidden" or not, Really)

- Often sources of valuable information
- Often hidden behind search interfaces
- Often non-crawlable by traditional crawlers



Interacting With Searchable Text Databases

- Searching: Metasearchers
- Browsing: Yahoo!-like directories
  - InvisibleWeb.com
  - SearchEngineGuide.com

Created Manually

Health > **Publications** > **PubMed**

How to Classify Text Databases Automatically

- Task definition
- Classification through query probing
- Experimental evaluation

ACM SIGMOD'01
ACM TOIS'03

Text Database Classification: Two Possibilities

- **Coverage**-based classification:
  - Database contains **many documents** about category
  - **Coverage**: #docs about this category
- **Specificity**-based classification:
  - Database contains **mainly documents** about category
  - **Specificity**: #docs/|DB|

Text Database Classification: An Example

Category: Basketball

- **Coverage-based** classification
  - ESPN.com, NBA.com
- **Specificity-based** classification
  - NBA.com, but **not** ESPN.com

Text Database Classification: More Details

- Define two "editorial" thresholds:
  - ***Tc:*** coverage threshold (# docs in category)
  - ***Ts:*** specificity threshold (fraction docs in category)

- Assign a text database to a category *C* if:
  - Database coverage for C at least ***Tc***
  - Database specificity for C at least ***Ts***

Brute-Force Database Classification “Strategy”

1. Extract all documents from database.
2. Classify documents.
3. Classify database accordingly.

Problem: No access to full contents of hidden-web databases!

Solution: Exploit database search interface to approximate document classification

Search-based Hidden-Web Database Classification

1. Train a (rule-based) document classifier.
2. Transform classifier rules into queries.
3. Adaptively send queries to databases.
4. Categorize databases based on adjusted number of query matches.

Training a Document Classifier

- Feature Selection: Zipf's law pruning, followed by information theoretic feature selection [Koller & Sahami'96]
- Classifier Learning: RIPPER [Cohen'95]
  - Input: A set of pre-classified, labeled documents
  - Output: A set of classification rules
    - IF linux THEN **Computers**
    - IF jordan AND bulls THEN **Sports**
    - IF heart AND pressure THEN **Health**

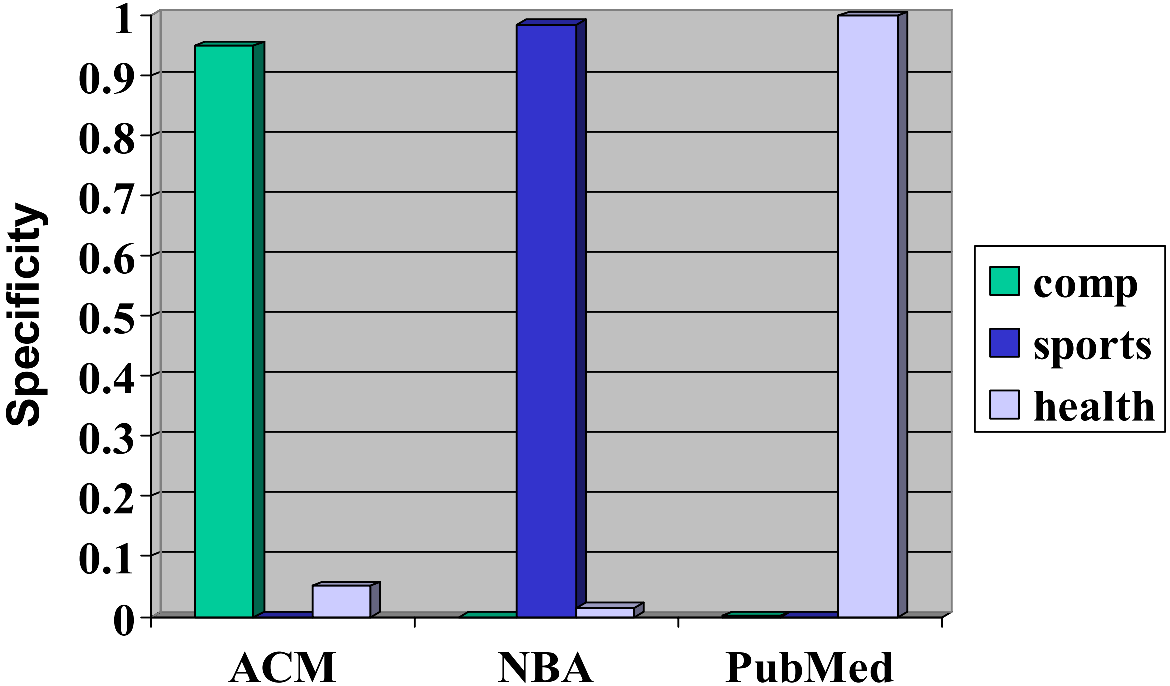
Designing and Implementing Query Probes

- Transform each document classifier rule into query:

  IF jordan AND bulls THEN Sports → +jordan +bulls

- Issue each query to database to obtain number of matches without retrieving any documents

Using Probe Results for Classification

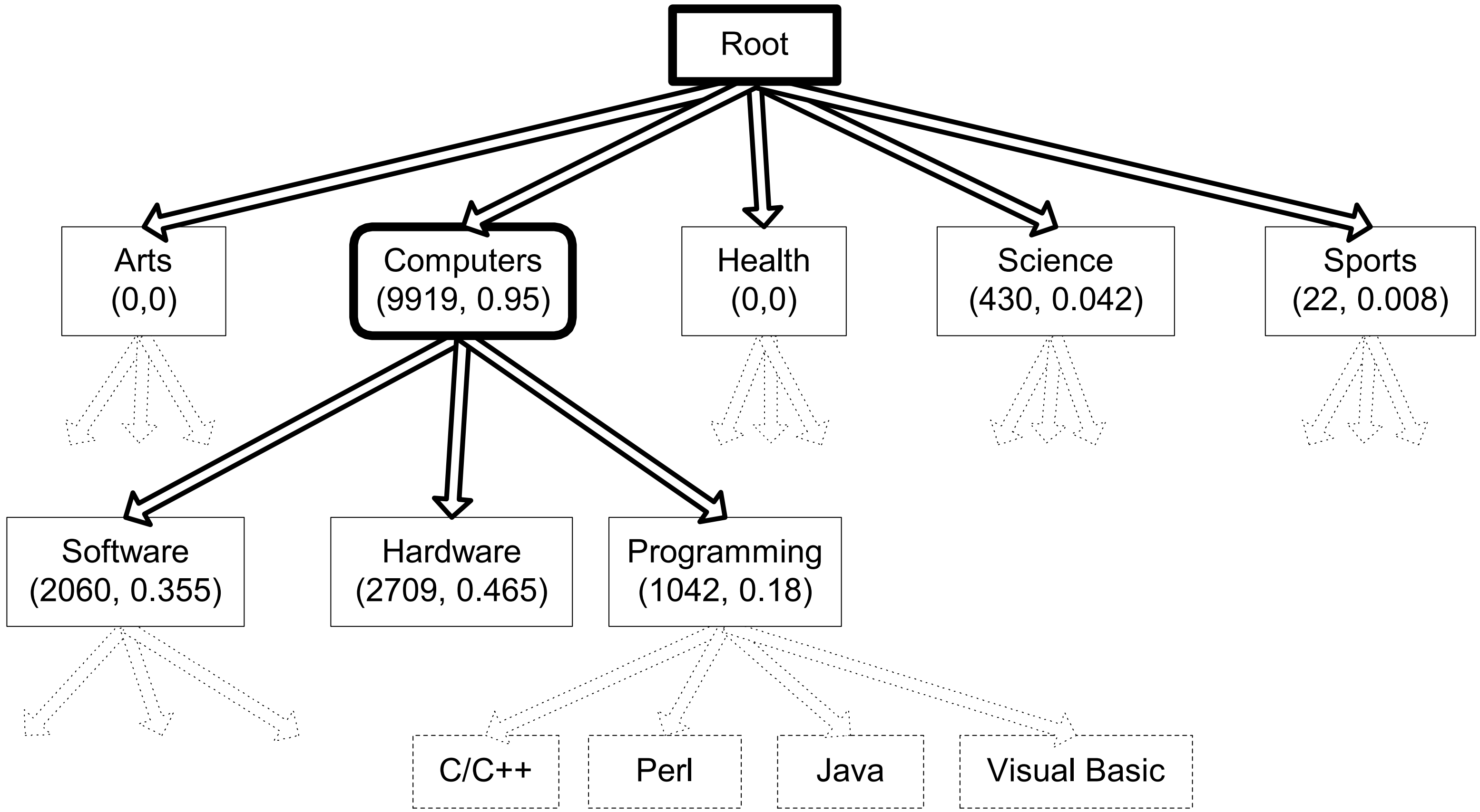


We use probe results to **estimate coverage** and **specificity**

COV	ACM	NBA	PubM
comp	336	0	16
sports	0	6674	0
health	18	103	81164

SPEC	ACM	NBA	PubM
comp	0.95	0	0
sports	0	0.985	0
health	0.05	0.015	1.0

Hierarchically Classifying the ACM DigLib (Tc=100, Ts=0.5)



Adjusting Query Results

- Search-based estimates of category distribution not perfect:
  - Queries for one category match documents from other categories
  - Queries might overlap
- Document classifiers not perfect:
  - Queries do not match all documents in a category

After classifier training, construct a **confusion matrix** for query probes

Confusion Matrix Adjustment of Query Probe Results

correct class

	comp	sports	health
comp	0.80	0.10	0.00
sports	0.18	0.85	0.04
health	0.02	0.05	0.96

assigned class

X

Real Coverage
1000
5000
50

=

Estimated Coverage
800+500+0 = 1300
180+4250+2 = 4432
20+250+48 = 318

This "multiplication" can be inverted to get the real coverage figures from the probe estimates.

Confusion Matrix Adjustment for Noise Reduction

$$\boxed{\begin{array}{c}\mathbf{M \,.\, Coverage(D) \sim ECoverage(D)} \\ \mathbf{Coverage(D) \sim M^{-1} \,.\, ECoverage(D)}\end{array}}$$

- *M* usually diagonally dominant for "reasonable" document classifiers, hence invertible
- Compensates for errors in search-based estimates of category distribution

Experiments: Data

- 72-node 4-level topic hierarchy from InvisibleWeb/Yahoo! (54 leaf nodes)
- 500,000 Usenet articles (April-May 2000):
  - Newsgroups assigned by hand to hierarchy nodes
  - RIPPER trained with 54,000 articles (1,000 articles per leaf)
  - 27,000 articles used to construct confusion matrix
  - Remaining 419,000 articles used to build **Controlled Databases**

Experiments: Data
Controlled Databases

500 databases built using 419,000 newsgroup articles

- One label per document
- 350 databases with single (not necessarily leaf) category
- 150 databases with varying category mixes
- Database size ranges from 25 to 25,000 articles
- Indexed and queries using SMART

Experiments: Data
Web Databases

- 130 real databases picked from InvisibleWeb (first 5 under each topic)
  - CancerBACUP; Iweb category: Cancer
  - Java@Sun; Iweb category: Java
  - John Hopkins AIDS Service; Iweb category: AIDS
- Only 12 with "newsgroup-like" data
- Used InvisibleWeb's categorization as correct
- Built simple "wrappers" for querying

Experimental Results: Controlled Databases

- **Feature selection helps.**
- **Confusion-matrix adjustment helps.**
- **F-measure above 0.8** for most <*Tc*, *Ts*> combinations tried.
- Results degrade gracefully with hierarchy depth.
- Relatively **small number of probes** needed for most <*Tc*, *Ts*> combinations tried.
- Also, **probes are short**: 1.5 words on average; 4 words maximum.

Experimental Results: Web Databases

- **F-measure above 0.7** for best <*Tc*, *Ts*> combination found.
- **185 query probes per database on average** needed for choice of thresholds.
- Also, **probes are short**: 1.5 words on average; 4 words maximum.

What if a "Database" is Crawlable?

1. Train a document classifier.
2. Using a crawler, download all documents from the web database.
3. Classify each document using the document classifier from Step 1.
4. Classify the database based on number of documents in each category.

Data Engineering Bulletin, March 2002

Crawling- vs. Query-based Classification for 5 Databases

URL	Brief Description	Category
http://www.cnnsi.com/	CNN Sports Illustrated	Sports
http://www.tomshardware.com/	Tom's Hardware Guide	Computers
http://hopkins-aids.edu/	Johns Hopkins AIDS Service	AIDS
http://odyssey.lib.duke.edu/	Duke University Rare Books	Literature
http://www.osti.gov/	Office of Scientific and Technical Information	Science

	Crawling-based Classification			*Query-based Classification*		
Database	*Time*	*Files*	*Size*	*Time*	*Queries*	*Size*
CNN Sports Illustrated	1325 min	270,202	8 Gb	2 min (-99.8%)	112	357 Kb (-99.9%)
Tom's Hardware Guide	32 min	2,928	105 Mb	3 min (-90.6%)	292	602 Kb (-99.7%)
Johns Hopkins AIDS Service	13 min	1,823	17 Mb	1 min (-92.3%)	314	723 Kb (-95.7%)
Duke University Rare Books	2 min	3,242	16.5 Mb	3 min (+50.0%)	397	1012 Kb (-93.8%)
Office of Scientific and Technical Information	210 min	30,749	416 Mb	2 min (-99.0%)	174	423 Kb (-99.8%)

Stability of Classification as Crawling Progresses

% Crawled	**10%**	**30%**	**50%**	**60%**	**70%**	**100%**
CNN Sports Illustrated	Cycling Multimedia $P = 0.5$ $R = 0.09$	Cycling Multimedia $P = 0.5$ $R = 0.09$	Cycling Multimedia $P = 0.5$ $R = 0.09$	Cycling $P = 1.0$ $R = 0.09$	**Sports** $P = 1.0$ $R = 1.0$	**Sports** $P = 1.0$ $R = 1.0$
Tom's Hardware Guide	**Computers** **Rock** $P = 0.91$ $R = 1.0$	**Computers** **Rock** $P = 0.91$ $R = 1.0$	**Computers** **Rock** $P = 0.91$ $R = 1.0$	**Computers** **Rock** $P = 0.91$ $R = 1.0$	**Computers** **Rock** $P = 0.91$ $R = 1.0$	**Computers** **Rock** $P = 0.91$ $R = 1.0$
Johns Hopkins AIDS Service	**AIDS** $P = 1.0$ $R = 1.0$	**AIDS** $P = 1.0$ $R = 1.0$	**AIDS** $P = 1.0$ $R = 1.0$	**AIDS** $P = 1.0$ $R = 1.0$	**AIDS** $P = 1.0$ $R = 1.0$	**AIDS** $P = 1.0$ $R = 1.0$
Duke University Rare Books	Poetry Texts Classics History Photography $P = 0.6$ $R = 0.6$	Poetry $P = 1.0$ $R = 0.2$	**Poetry** **Texts** $P = 1.0$ $R = 0.4$	**Poetry** **Texts** $P = 1.0$ $R = 0.4$	**Poetry** **Texts** $P = 1.0$ $R = 0.4$	**Poetry** **Texts** $P = 1.0$ $R = 0.4$
Office of Scientific and Technical Information	Biology $P = 1.0$ $R = 0.33$	Root $P = 0.25$ $R = 1.0$	Root $P = 0.25$ $R = 1.0$	**Biology** $P = 1.0$ $R = 0.33$	**Biology** $P = 1.0$ $R = 0.33$	**Biology** $P = 1.0$ $R = 0.33$

Beyond Original Setting

- Adapting reformulation to other search-engine interfaces (e.g., Boolean vs. vector-space)
- Exploiting other document classifiers
  - Not rule-based: SVMs, Bayesian models, C4.5
  - Rules extracted from learned models

ACM TOIS 2003

Query-based Database Classification

- Easy classification using just a few queries
- No need for document retrieval
  - Only need to identify a line like: “82 matches found”
  - “Wrapper” needed is trivial
- Not limited to hidden-web databases: query-based approach sometimes orders of magnitude more efficient than crawling

Outline of Talk

- Classification of Hidden-Web Databases
- **Search over Hidden-Web Databases**
- Overview of Columbia's Database Group

Interacting With Searchable Text Databases

- **Searching: Metasearchers**
- Browsing: Yahoo!-like directories
  - InvisibleWeb.com
  - SearchEngineGuide.com

Health > Publications > PubMed

Three Main Metasearcher Tasks

- Database Selection:
  Choosing best databases for a query
- Query Translation:
  Translating query for each chosen database
- Result Merging:
  Combining query results from chosen databases

Database Selection Step Needs Database "Content Summaries"

Typically the vocabulary of each database plus simple frequency statistics:

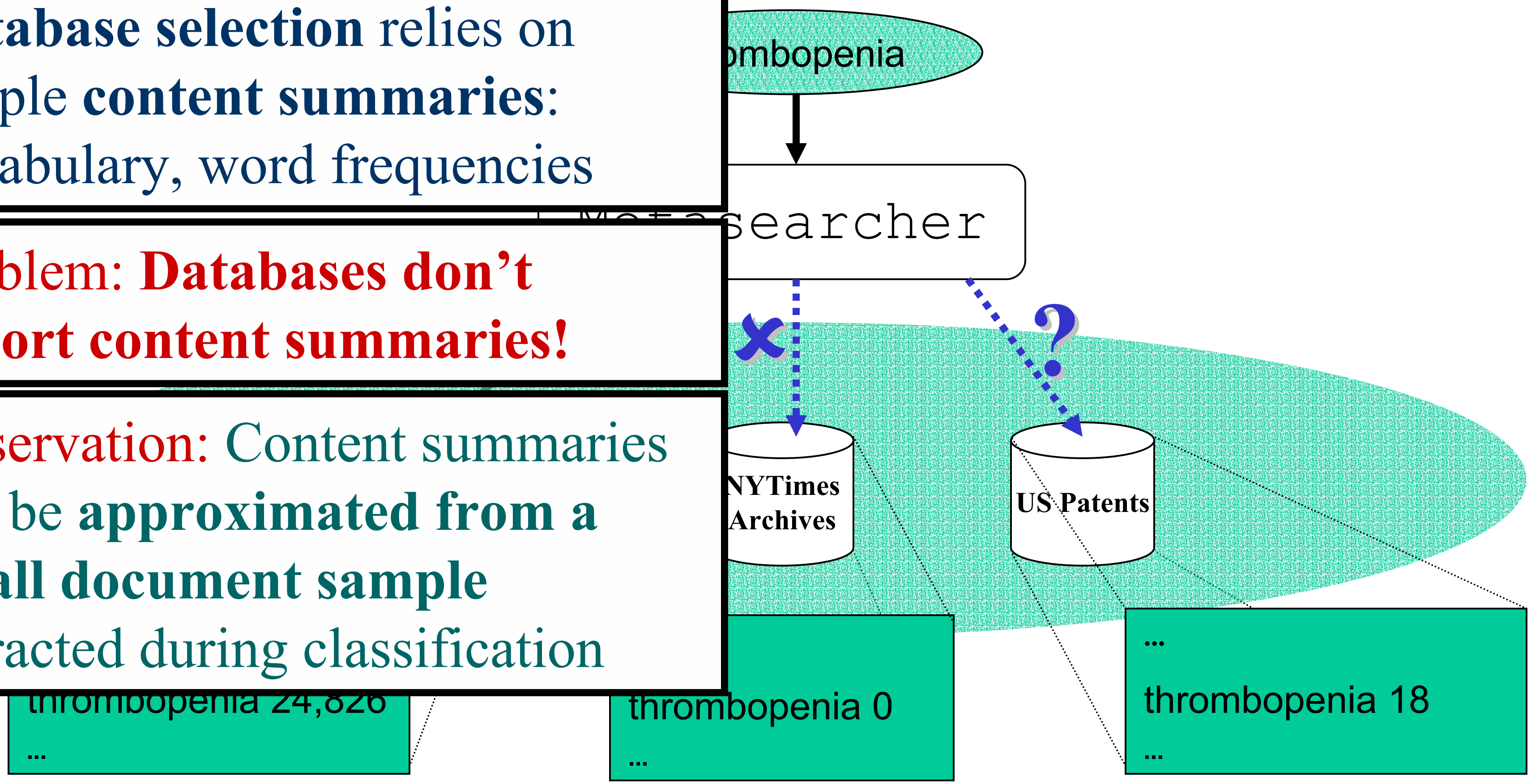
PubMed (3,868,552 documents)	
…	
cancer	1,398,178
aids	106,512
heart	281,506
hepatitis	23,481
thrombopenia	24,826
…	

Metasearchers Provide Access to Distributed Databases

Database selection relies on simple **content summaries**: vocabulary, word frequencies

Problem: **Databases don't export content summaries!**

Observation: Content summaries can be **approximated from a small document sample** extracted during classification



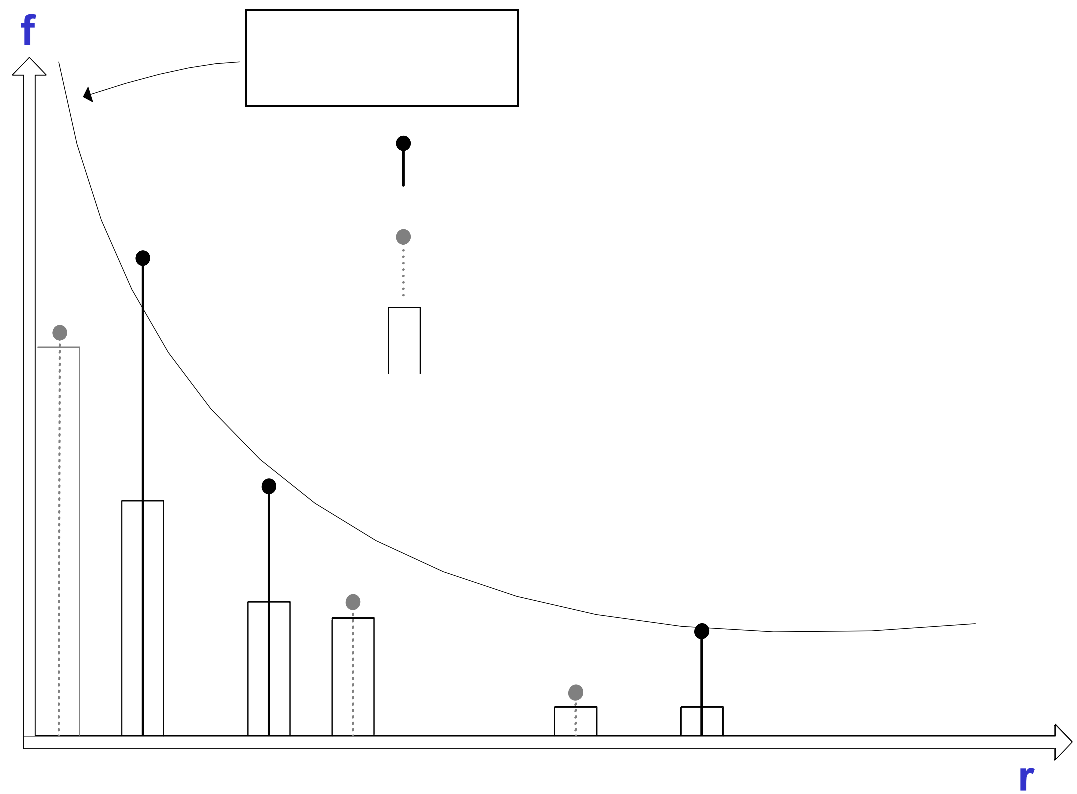
Extracting a Document Sample for Content Summary Construction

1. Train a (rule-based) document classifier.
2. Transform classifier rules into queries.
3. Adaptively send queries to databases.
   - **Retrieve top-*k* matching documents for each query.**
   - **Save #matches for each one-word query.**
4. Categorize databases based on number of query matches.

Output:
- **Representative document sample**
- **Actual frequencies for some "important" words, from queries**

Adjusting Document Frequencies

- We know ranking r of words according to document frequency in sample
- We know absolute document frequency f of some words from *one-word queries*
- Mandelbrot's formula connects empirically word frequency f and ranking r
- We use curve-fitting to estimate the absolute frequency of **all words** in sample



Actual PubMed Content Summary

PubMed (3,868,552 documents)
Categories: Health, Diseases

…

cancer	1,398,178
aids	106,512
heart	281,506
hepatitis	23,481
…	
basketball	907
cpu	487

- Extracted **automatically**
- ~ **27,500 words** in extracted content summary
- **Fewer than 200** queries sent
- **At most 4 documents** retrieved per query

(heart, hepatitis, basketball not in 1-word probes)

Database Selection and Extracted Content Summaries

- Database selection algorithms assume complete content summaries
- Content summaries extracted by (small-scale) sampling are inherently incomplete (Zipf's law)
- Queries with undiscovered words are problematic

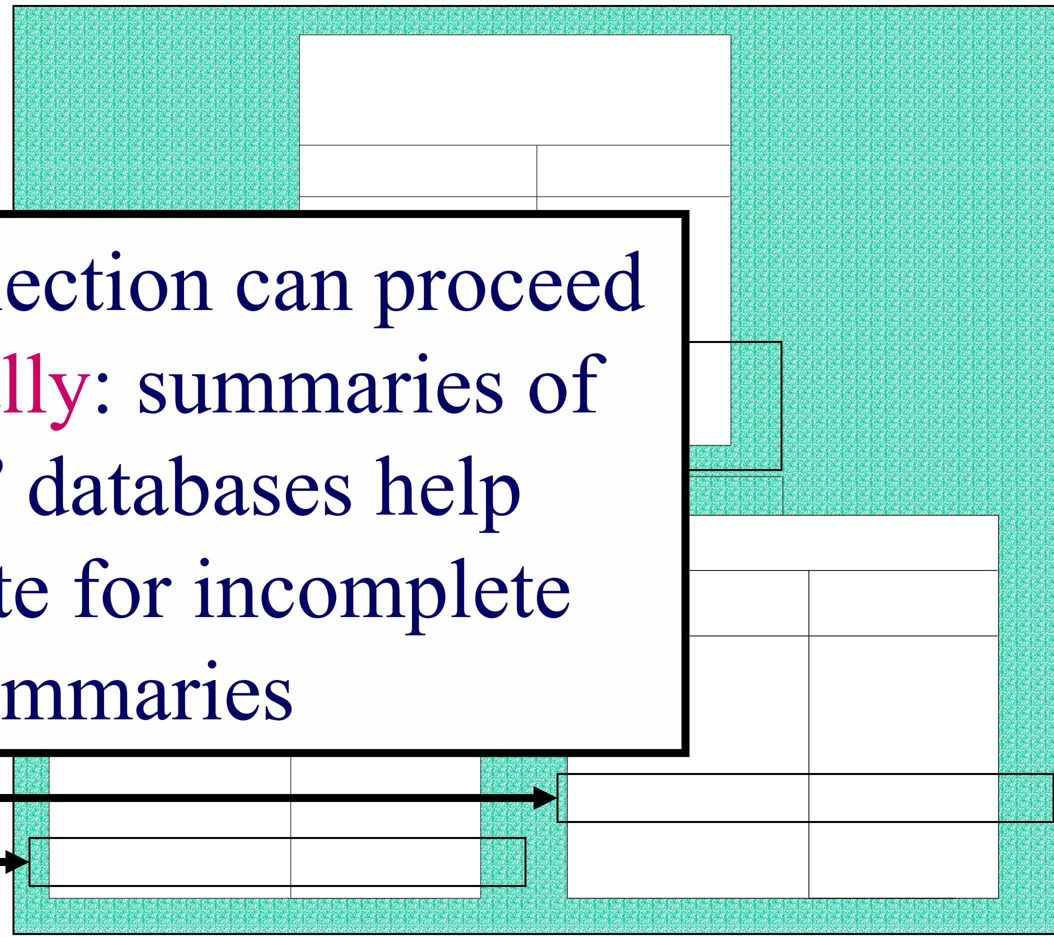
Database Classification Helps:

Similar topics ↔ Similar content summaries

Extracted content summaries complement each other

Content Summaries within Category Complement Each Other

- Cancerlit contains "**thrombopenia**", not found d
- PubMed "**chemo** found d
- **Health** summar

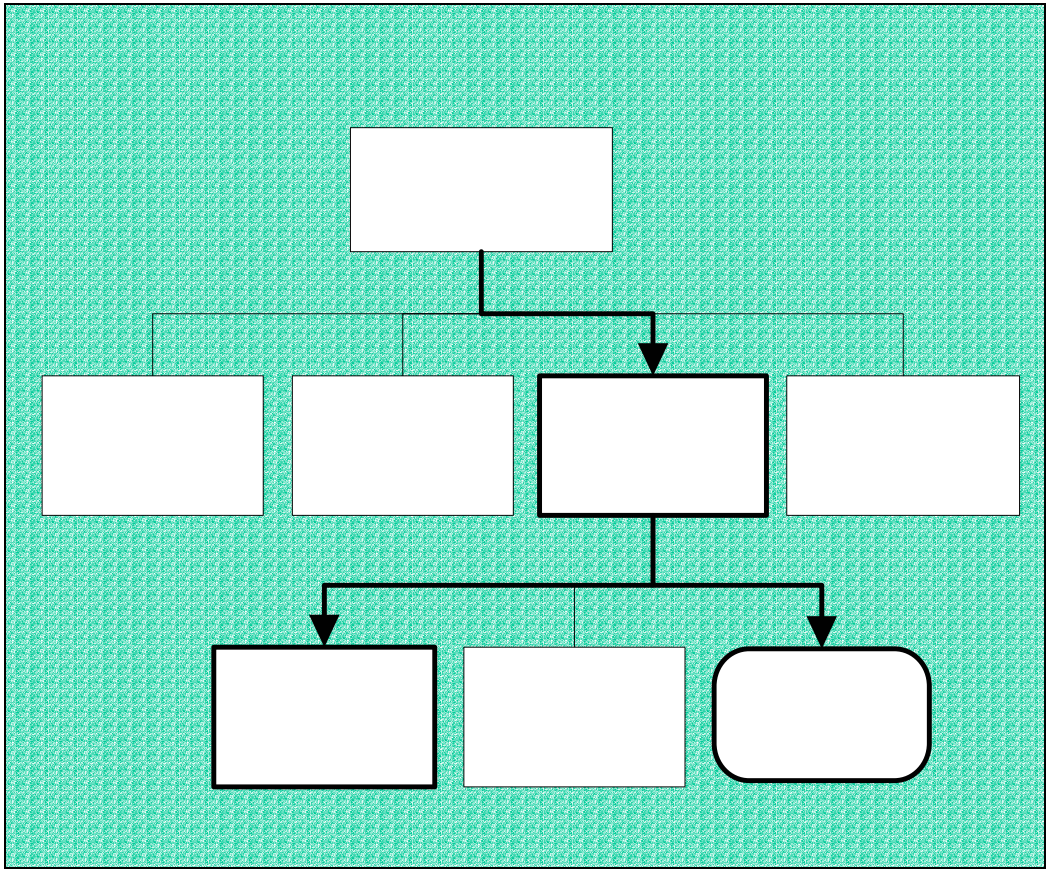


Database selection can proceed hierarchically: summaries of "sibling" databases help compensate for incomplete summaries

Hierarchical DB Selection: Example

To select **D** databases:

1. Use "flat" DB selection algorithm to score categories
2. Proceed to category with highest score
3. Repeat until category is a leaf, or category has fewer than **D** databases



Hierarchical Hidden-Web Database Sampling and Selection

- We extract content summaries efficiently from "uncooperative" hidden-web databases
- We estimate absolute word frequencies
- We improve effectiveness of hierarchical database selection by exploiting database classification

Content summary extraction code available at:
http://sdarts.cs.columbia.edu

Outline of Talk

- Classification of Hidden-Web Databases
- Search over Hidden-Web Databases
- **Overview of Columbia's Database Group**

My Columbia Database "Sub-group"

Ph.D. Students

Eugene Agichtein
Nicolas Bruno
Wisam Dakka
Panos Ipeirotis
Amélie Marian

Faculty

Luis Gravano

Some Themes

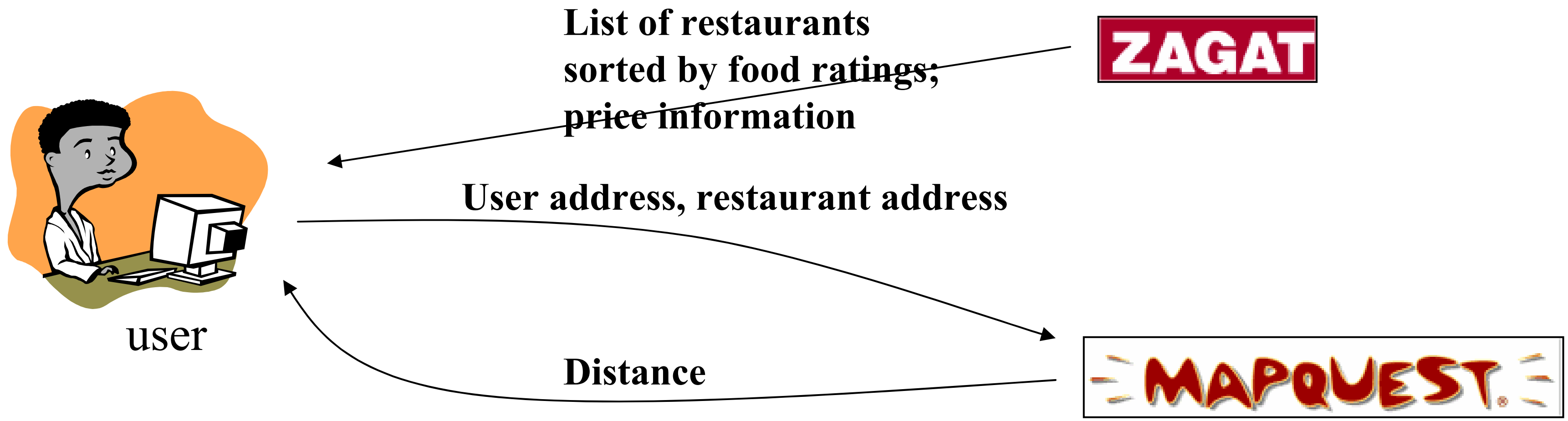
- "Top-*k*" query processing
- Information extraction from web resources
- (Distributed) web search
- Web "mining"
- …

“Top-*k*” Query Processing over Web-Accessible Sources – Amélie Marian

Top-*k* Query: Specification of (flexible) preferences

“Italian restaurants near my home for <$25”

Answer: Best *k* answers according to distance function



- Goal is to minimize number of remote queries.
- A challenge is to handle different source access capabilities.

IEEE ICDE’02

Efficient Information Extraction with Minimal Training – Eugene Agichtein

Microsoft's central headquarters in *Redmond* is home to almost every product group and division.

Brent Barlow, 27, a software analyst and beta-tester at *Apple Computer*'s headquarters in *Cupertino*, was fired Monday for "thinking a little too different."

Apple's programmers "think different" on a "campus" in *Cupertino, Cal*. *Nike* employees "just do it" at what the company refers to as its "World Campus," near *Portland, Ore*.

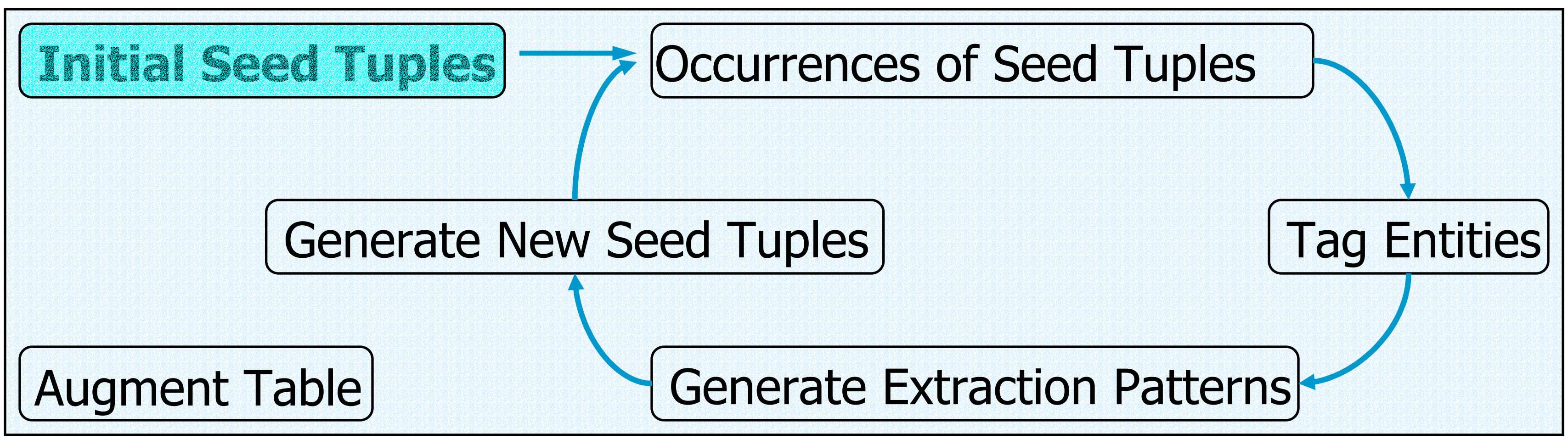
Organization	Location
Microsoft	Redmond
Apple Computer	Cupertino
Nike	Portland

Extracting Relations from Text: **Snowball**

- Exploit redundancy on web to focus on "easy" instances
- Require only minimal training (handful of seed tuples)

ORGANIZATION	***LOCATION***
MICROSOFT	REDMOND
IBM	ARMONK
BOEING	SEATTLE
INTEL	SANTA CLARA

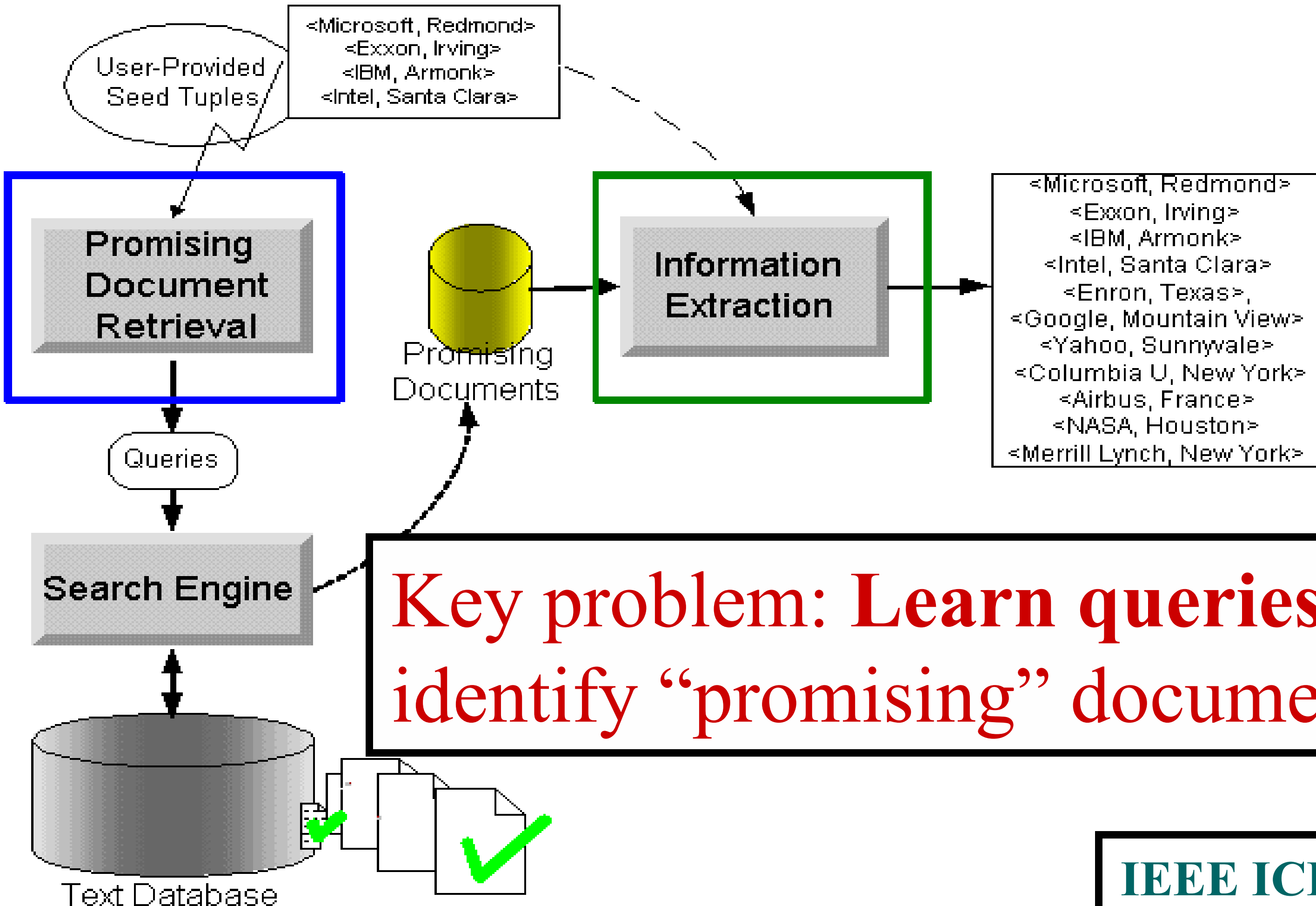
ACM DL'00



Information Extraction is Expensive

- Efficiency is a problem even after information extraction system is trained
- Example: NYU's Proteus extraction system takes around 7 seconds per document
- Can't afford to "scan the web" to process each page!

Querying For Efficient Extraction: **QXtract**



Key problem: **Learn queries** to identify "promising" documents

IEEE ICDE'03

"Sub-Expression Statistics" in Relational Query Optimization – Nico Bruno

- Relational query optimizers rely on cost estimates to choose query execution plans.
- Plan costs heavily influenced by cardinalities of sub-expressions of queries.
- Optimizers estimate such cardinalities using simplifying assumptions.

Approach: Identify "sub-expression statistics" to maintain and incorporate into query optimization

ACM SIGMOD'02, IEEE ICDE'03

Some Links

http://www.cs.columbia.edu/~gravano

- **Snowball**, an information-extraction system
  http://snowball.cs.columbia.edu
- **QProber**, a system for classifying and searching "hidden-web" text databases
  http://qprober.cs.columbia.edu
- **SDARTS**, a protocol and toolkit for metasearching
  http://sdarts.cs.columbia.edu
- **RANK:** top-*k* query processing
  http://rank.cs.columbia.edu
- **PERSIVAL**, personalized search and summarization over multimedia information
  http://persival.cs.columbia.edu