Searching the Deep Web

1

What is Deep Web?

- Information accessed *only* through HTML form pages
  - database queries
  - results embedded in HTML pages
- Also can included other information on Web can't directly index
  - Javascript output
    - simulate, extract info from results?
  - unlabeled images, video, music, …
  - password protected Web pages
- compare invisible Web
  - pages on servers with no paths from crawler seeds

2

Extent of problem

- Estimates
  - 500 times larger than "surface" Web in terabytes of information
  - diverse uses and topics
    - 51% databases of Web pages behind query forms non-commercial (2004)
      - includes pages also reachable by standard crawling
    - 17% surface Web sites are not commercial sites (2004)
  - in 2004 Google and Yahoo each indexed 32% Web objects behind query forms
    - 84% overlap ⇒ 63% not indexed by either

3

Growth estimates

- 43,000-96,000 Deep Web sites est. in 2000
  - 7500 terabytes ⇒ 500 times surface Web
  - estimate by overlap analysis - underestimates
- 307,000 Deep Web sites est. 2004 (2007 CACM)
  - 450,000 Web databases: avg. 1.5 per site
  - 1,258,000 unique Web query interfaces (forms)
    - avg. 2.8 per database
    - 72% at depth 3 or less
    - 94% databases have some interface at depth 3 or less
    - exclude non-query forms, site search
  - estimate extrapolation from sampling

4

Random sampling

- are 2,230,124,544 valid IPv4 addresses
- randomly sample 1 million of these
- take 100,000 IP address sub-sample
- For sub-sample
  - make HTTP connection & determine if Web server
  - crawl Web servers to depth 10
- For full sample
  - make HTTP connection & determine if Web server
  - crawl Web servers to depth 3

5

Analysis of data from samples

- Find
  - # unique query interfaces for site
  - # Web databases
    - query interface to see if uses same database
  - # deep Web sites
    - not include forms that are site searches
- Extrapolate to entire IP address space

6

Approaches to getting deep Web data

- Application programming interfaces
  - allow search engines get at data
  - a few popular site provide
  - not unified interfaces
- virtual data integration
  - a.k.a. mediating
  - "broker" user query to relevant data sources
    - issue query real time
- Surfacing
  - a.k.a warehousing
  - build up HTML result pages in advance

7

Virtual Data Integration

- In advance:
  - identify pool of databases with HTML access pages
    - crawl
  - develop model and query mapping for each source: mediator system
    - domains + semantic models
    - identify content/topics of source
    - develop "wrappers" to "translate" queries

8

Virtual Data Integration

- When receive user query:
  - from pool choose set of database sources to query
    - based on source content and query content
    - real-time content/topic analysis of query
  - develop appropriate query for each data source
  - integrate (federate) results for user
    - extract info
    - combine (rank?) results

9

Mediated scheme

- Mappings

  form inputs → elements of mediated scheme

  query over mediated scheme

  → queries over each form

  user query → query over mediated scheme
- creating mediated scheme
  - manually
  - by analysis of forms HARD

10

Virtual Integration: Issues

- Good for specific domains
  - easier to do
  - viable when commercial value
- Doesn't scale well

11

Last time

- Intro to Deep Web issues
- Approaches to getting deep Web data
  - virtual integration

Today

- Approaches to getting deep Web data
  - Surfacing
- Semi-structured documents
  - XML

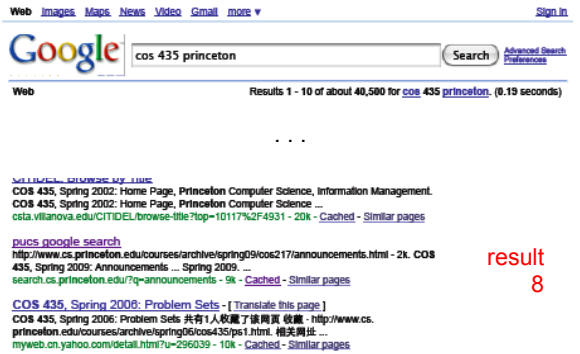
12

Surfacing

- In advance:
  - crawl for HTML pages containing forms that access databases
  - for each form
    - execute many queries to database using form
      - how choose queries?
    - index each resulting HTML page as part of general index of Web pages
  - pulls database information to surface
- When receive user query:
  - database results are returned like any other

Google query: cos 435 princeton

executed April 30, 2009 in AM



result 8

cached version of pucs google search

Surfacing: Google methodology

- Major Problem:
  Determine queries to use for each form
  - determine templates
  - generate values for selected inputs
- Goal:
  Good coverage of large number of databases
  - "Good", not exhaustive
    - limit load on target sites during indexing
    - limit size pressure on search engine index
    - want "surfaced" pages good for indexing
  - trading off depth within DB site for breadth of sites

Query Templates

- given form with n inputs
- choose subset of inputs to vary => template
  - choose from text boxes & select menues
    "state" select menu, "search box" text box, "year" select menu
  - values for choosen inputs will vary
  - rest of inputs set to defaults or "don't care"
  - want small number choosen inputs
    - yield smaller number form submissions to index

17

Building Query Templates

- Want "informative templates":
  when vary choosen input values,
  pages generated are "sufficiently distinct"
- Building informative templates
  - start with templates for single choosen input
  - repeat:
    - extend "informative templates" by 1 input
    - determine "informativeness" for each new template

18

Informative Templates

- Informative if generates "sufficiently distinct" pages
- use page signature for "informativeness" test
- Signatures create clusters pages
  - One cluster per signature
- Informative if
  (# clusters) / (# possible pages from template) exceeds a threshold

19

Generating values

generic text boxes: any words

for one box:

- select seed words from form page to start
- use each seed word as input to text box
- extract more keywords from results
  - tf-idf analysis
  - remove words occur in too many of pages in results
  - remove words occur in only 1 page of results
- repeat until no new keywords or reach max
- choose subset of keywords found

20

Generating values

choosing subset of words for generic boxes

- cluster keywords based on words on page generated by keyword
  - words on page characterize keyword
- choose 1 candidate keyword per cluster
- sort candidate keywords based on page length of form result
- choose keywords in decreasing page-length order until have desired number

21

Generating values

text boxes with fixed types: well-defined set values

- type can be recognized with high precision
  - relatively few types over many domains
    - zip code, date, …
  - often distinctive input names
  - test types using sample of values

22

Google designers' observations

- # URLs generated proportional to size database, not # possible queries
- semantics not "significant role" in form queries
  - exceptions: correlated inputs
    - min-max ranges - mine collection of forms for patterns
    - keyword+database selection - HARD when choice of databases (select box)
- user still gets fresh data
  - Search result gives URL with embedded DB query
    - doesn't work for POST forms

23

more observations

- became part of Google Search
  - in results of "more than 1000 queries per second" 2009
- impact of "long tail of queries"
  - top 10,000 forms acct for only 50% Deep Web results
  - top 100,000 forms acct for only 85% Deep Web results
  - Need surface as many sites as possible
- domain independent approach important
- wish to automatically extract database data (relational) from surfaced pages

24

Google deep web crawl for "entity pages"

- builds on work just seen
- simpler than that work - specialized
- entities versus text content
  - examples
    - products on shopping sites
    - movies on review sites
  - structured: well-defined attributes
- motivation
  - crawl product entities for advertisement use

25

Major steps: 1: URL template generation

- get list of "entity-oriented deep-web sites"
- extract search forms
  - usually home page
- produce one template per search form
  - observe usually one main text input field
  - set other fields to default
    - observe get "good behavior" doing this

26

Major steps, 2: Query generation

find query words to use in main text field

- use Google query log for site for candidates
  - site URL clicked? How many times?
- isolate entity keywords from queries
  - example: "HP touchpad ~~reviews~~"
  - identify common patterns to remove
    - analyze query logs using known entities

      Freebase –"community curated" entity keywords
- expand using Freebase
  - Freebase entities organized by domain/category

27

Next challenges

- Web is MUCH more dynamic than when most of work we've discussed was done and much more interactive
- Other challenges to further extend ability to extract and organize data:
  - Automatically extract data from general pages
  - Combining data from multiple sources
    - general, not custom, solution

28