Searching the Deep Web

What is Deep Web?

- Information accessed only through HTML form pages
  - database queries
  - results embedded in HTML pages
- (was) part of invisible Web
  - any information on Web can’t search
    - Javascript output
    - unlabeled images, video, music, …
    - extract information?
  - pages sitting on servers with no paths from crawler seeds

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

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

Random sampling

- are 2,230,124,544 valid IP 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

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
# 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

---

## 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

## 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

---

## Mediated scheme

- **Mappings**
  - form inputs → elements of mediated scheme
  - query over mediated scheme
    → queries over each form

- **Creating mediated scheme**
  - manually
  - by analysis of forms **HARD**

---

## Virtual Integration: Issues

- Good for specific domains
  - easier to do
  - viable when commercial value

- Doesn’t scale well

---

## 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 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

---

**Surfacing: Google methodology**

- **Major Problem:**
  - Determine queries to use for each form
  - determine templates
    - SELECT * FROM DB WHERE predicates
  - generate values for predicates

- **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

---

**Google: Query Templates**

- form with n inputs
- designate subset of inputs as “binding”, rest free
  - binding inputs from text boxes & select menus
  - values for binding inputs will vary, giving predicates
  - free inputs set to defaults or “don’t care”
  - want small number binding inputs
    - yield smaller number form submissions to index
  - start with templates for single binding inputs
  - repeat: extend “informative templates” by 1 input
    - “informative” = pages generated using different values for binding inputs are sufficiently distinct

---

**Google: generating values**

- generic text boxes: any words
  - for one box
  - select seed words from form page to start
  - use each seed word as inputs 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

---

**Google: 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
Google: generating values

typed text boxes: 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

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

more observations

• is now part of Google Search
  – in results of “more than 1000 queries per second” 2009
• impact on “long tail of queries”
  – top 10,000 forms acct for 50% Deep Web results
  – top 100,000 forms acct for 85% Deep Web results
• domain independent approach important
• next (now?) automatically extract database data
  (relational) from surfaced pages

Univ Utah DeepPeep

• specializes in Web forms
• goal: index all Web forms
• “tracks 45,000 forms across 7 domains”
• claims 90% content retrieved each indexed site
• uses focused crawler
• http://www.deeppeep.org/

Deep Peep focused crawler

• Classifiers
  – Pages classified by taxonomy
    e.g. arts, movies, jobs, ....
  – Form classifier
  – Link classifier
    • Want links likely lead to search form interfaces
    eventually
  • Learn features of good paths
    – Get samples by backwards crawls
    – words in neighborhood of links are features
      for training: URL, anchor text, nearby text

Deep Web: Related Problems

• Extract data from HTML tables
  – turn into database tables
• Extract information from HTML lists

• Applications
  – search databases
    • return databases not pages
  – question answering
  – aggregating information
    • mashups
Google WebTables

• Find relational HTML tables
  – about 1% of all HTML tables
  – step 1: throw out obvious non-relational
    • use hand-written detectors
      – single row or column
      – calendars
      – HTML form layout
    • throws out >89% of tables found in crawl

Google WebTables: 2008 results

• crawled “several billion” Web pages
• estimate 154 million true relations
• Step 2 finds 271 million relations
• estimate 125 million found relations are true relations
  – 81% of all true relations
  – 46% of all relations found

Google WebTables: Find relational HTML tables, cont.

• Step 2: use statistical classifier
  – labels relational or non-relational
  – hand-written features, for example:
    • each column uniform data type?
    • few empty cells?
  – train on human-judged sample
• Step 3: recover metadata
  – limit to column labels
  – use trained classifier: has metadata or not

Next challenges

• Data behind Javascript code
  – mashups, visualizations
• Combining data from multiple sources
  – general, not custom, solution