

Project 2

Advanced Databases

Amandeep Singh
as3947@columbia.edu

Evangelia Sitaridi
es2996@cs.columbia.edu

March 22, 2011

1 Description

For classification and content summary construction we used the fixed classification tree of figure 1, given in the description of the project. The queries used for the classification are in directory queries, included in the submission compressed file. Our implementation has two main classes shown in 2.

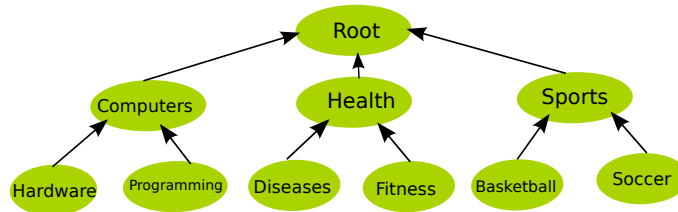


Figure 1: Category tree

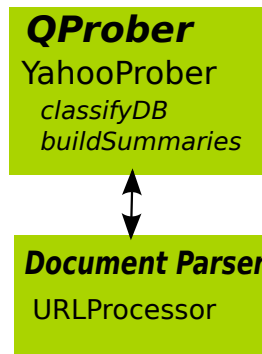


Figure 2: Implementation Classes

1.1 Classification

To classify the given database we traverse each level of the category tree and compute the following metrics for each category:

$$ECoverage(D, C_i) = \sum_{q: \text{query probe for } C_i} f(q)$$

$$ESpecificity(D, C_i) = \frac{ESpecificity(D, Parent(C_i)) \times ECoverage(D, C_i)}{\sum_{C_j \text{ is a child of } Parent(C_i)} ECoverage(D, C_j)}$$

pruning the categories that do not meet the specificity and coverage criteria. We implemented an iterative version of QProber [2]. The pseudocode is provided below.

```

Classify(Ts, Tc)
  Cat={Root}
  insert Root to L
  P=Root
  level=0
  do{
    for each category P in L
      read P query-file F
      add to L children of P
      for each query q in F
        pose-query(q)
        store top-4 results
        update coverage of P
        update coverage of q.child
      if(P.specificity>=Ts && P.coverage>=Tc)
        add P to Cat
  }while(levels<=2)
  return Cat

```

1.2 Content Summary Construction

For this step we just parse the top-documents fetched during query-probes from the classification step for each of the classified categories. We eliminate duplicates before constructing a content-summary. We also decided not to include multiple-word entries to the content summaries. To space the requests to the web-sites we process first parse the fetched document and then we fetch the next url. To parse the documents we used a slight variation of the provided Java script.

1.3 Exception handling

There are two main types of exceptions:

- Connections timeouts when querying Yahoo (java.net.SocketTimeoutException)
- Http error (code 503)

In both cases we retry getting the results from Yahoo, up to a maximum number of tries, set to 100 which is more than enough.

URL	Description	Classification
java.sun.com	Java@Sun	Root/Computers/Programming
yahoo.com	Yahoo SE	Root
diabetes.org	American Diabetes Assoc.	Root/Health
tomshardware.com	Tom's Hardware	Root/Computers/Hardware
hardwarecentral.com	PC Hardware Reviews	Root/Computers
espn.com	WorldWide Leader in Sports	Root/Sports/Basketball
portal.acm.org	ACM Digital Library	Root/Computer/Programming
hopkins-aids.edu	HIV guide	Root/Health/Diseases
agiweb.org	American Geological Institute	Root/Health
www.cancer.gov	National Cancer Institute	Root/Health/Diseases
www.ncbi.nlm.nih.gov/PubMed	PubMed	Root/Health/Diseases
www.ovid.com/site/index.jsp	Ovid Technologies	Root/Health/Diseases
soccernet.espn.go.com	Football News & Scores	Root/Sports/Soccer
www.jumbo.com	Free Computer Software	Root/Computers/Programming
www.webmd.com	Medical News & Information	Root/Health
www.fitnessmagazine.com	Fitness Magazine	Root/Health/Fitness
www.afaa.com	Aerobic & Fitness Info of America	Root/Health/Fitness
processing.org	Processing Software	Root/Computer/Programming
www.telegraph.co.uk/sport	Telegraph Sport News	Root/Sports/Soccer
nba.com	National Basketball Association	Root/Sports/Basketball
www.sports.org.au	Australian Athletes with Disability	Root/Sports

Table 1: Classification for $t_s=0.3$ and $t_c=100$

2 Example

To test the classification results of QProber we used the provided the examples in the project description and several of the examples from a list of Largest Deep Web Sites [1]. The results 1 are for $T_s=0.6$ and coverage $T_c=100$ (computed on 20th of March). The specificity is ≈ 0.5 so all web-pages were classified under one category.

Appendix

File Listing

We list below the files included in our submission zip. The zip contains the following folders:

[src, lib, queries]. The files are:

- README.pdf
- src/YahooProber.java
- src/URLProcessor.java
- src/Makefile
- lib/json.jar

- queries/Root.txt
- queries/Computers.txt
- queries/Health.txt
- queries/Sports.txt

Compilation

- **Compile:** make
- **Clean-up:** make clean
- **Execute:** java -cp ../lib/json.jar YahooProber *< url >* *< t_{es} >* *< t_{ep} >* *< yahoo – app – id >*
- **Yahoo App Id:**
BEWTNqTV34H1zojJNQ5MZB48A1vR2mJeNAhKRvk5.bLyZd6gYgQmsVVsqZ7vv32aW73O6VNyzTO

References

- [1] *Largest Deep Web Sites*, http://aip.completeplanet.com/aip-engines/help/largest_engines.jsp
- [2] *QProber: A system for automatic classification of hidden-web databases*, Panagiotis G. Ipeirotis, Luis Gravano, Mehran Sahami, 2003