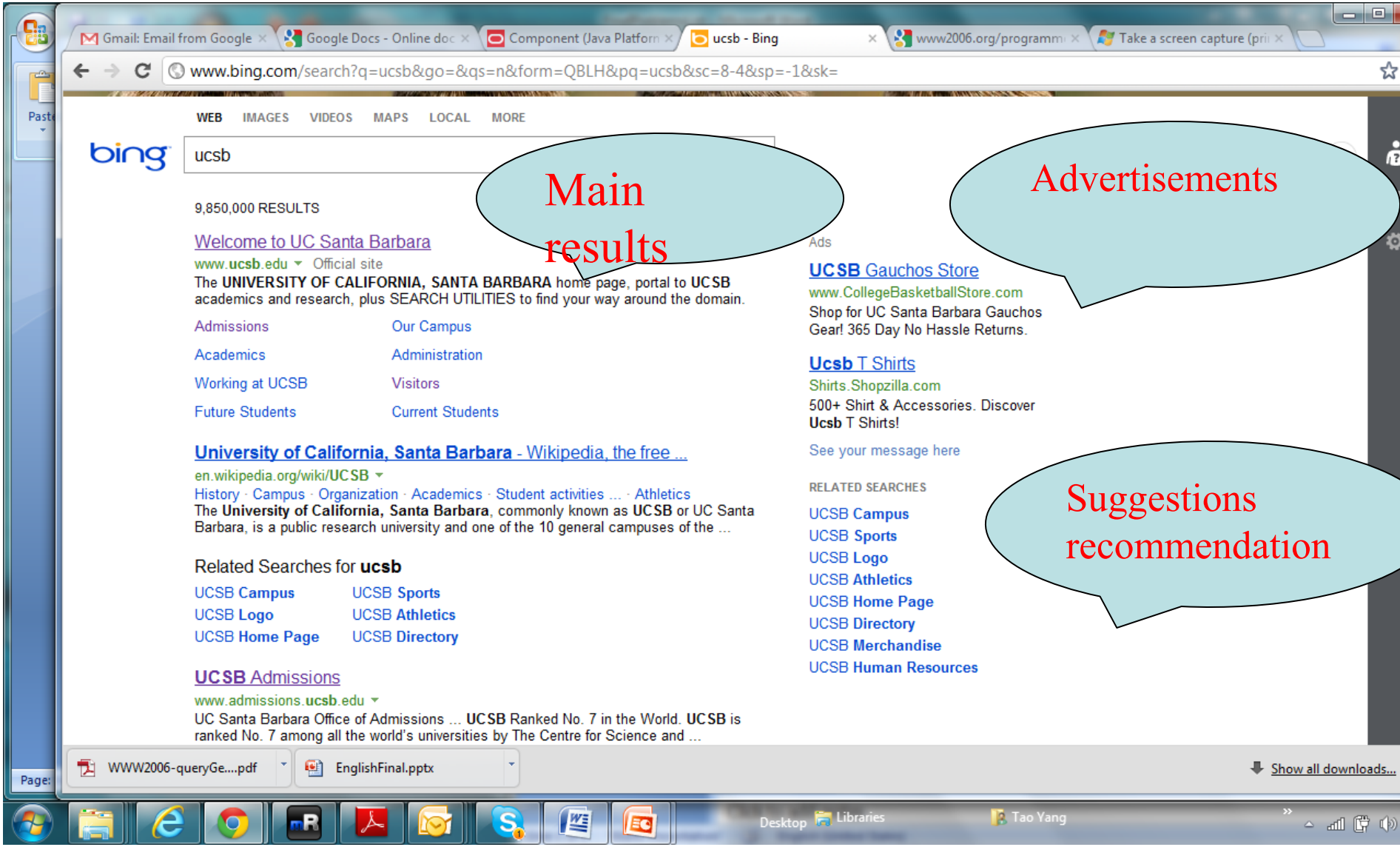
CS290N Summary

2015 Tao Yang

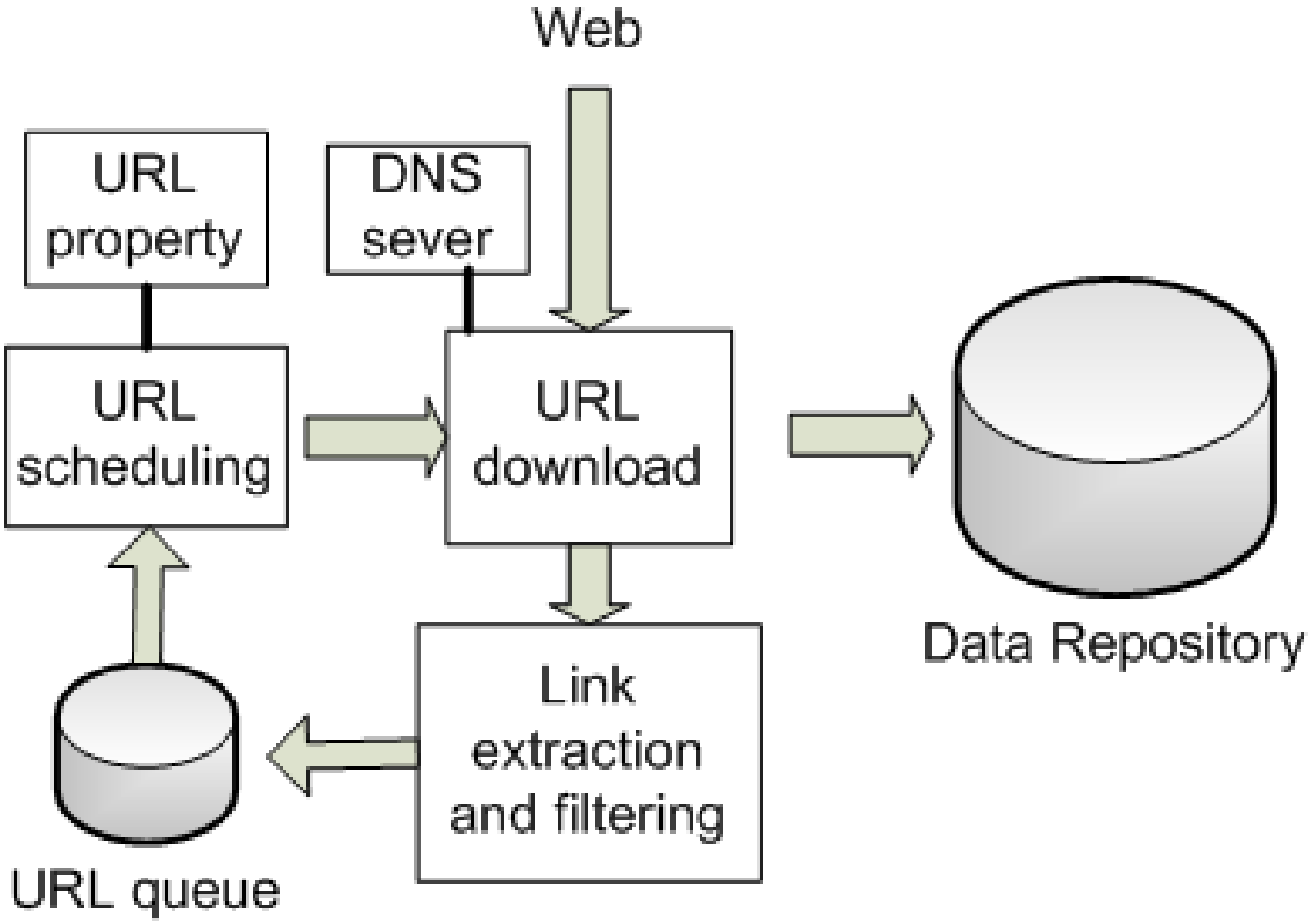
Text books

- [CMS] Bruce Croft, Donald Metzler, Trevor Strohman, Search Engines: Information Retrieval in Practice, Publisher: Addison-Wesley, 2010. Book website .
- [MRS] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, *Introduction to Information Retrieval*, Cambridge University Press. 2008. HTML edition of the book here.
- Ricardo Baeza-Yates and Berthier Ribeiro-Neto, Modern Information Retrieval (second edition), Addison-Wesley, 2011. Book website .
- Charles L. A. Clarke, Stefan Buettcher, Gordon V. Cormack, Information Retrieval: Implementing and Evaluating Search Engines, MIT Press Book website .

Search Result Reply Pages

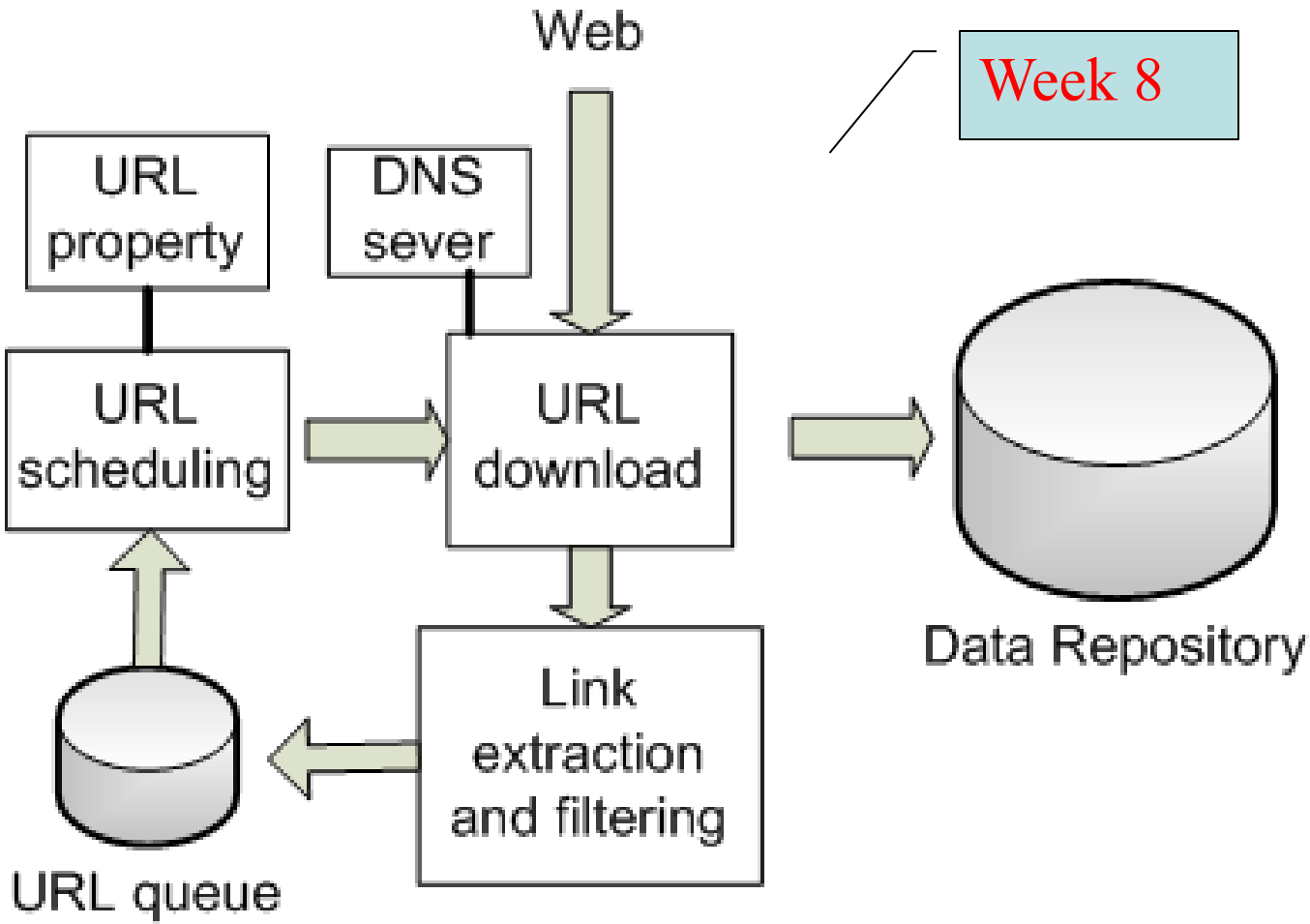


A Crawler Architecture



Olston/Najork. Web crawling.
Found. Trends Inf. Retr., 4(3):175--246, March 2010.

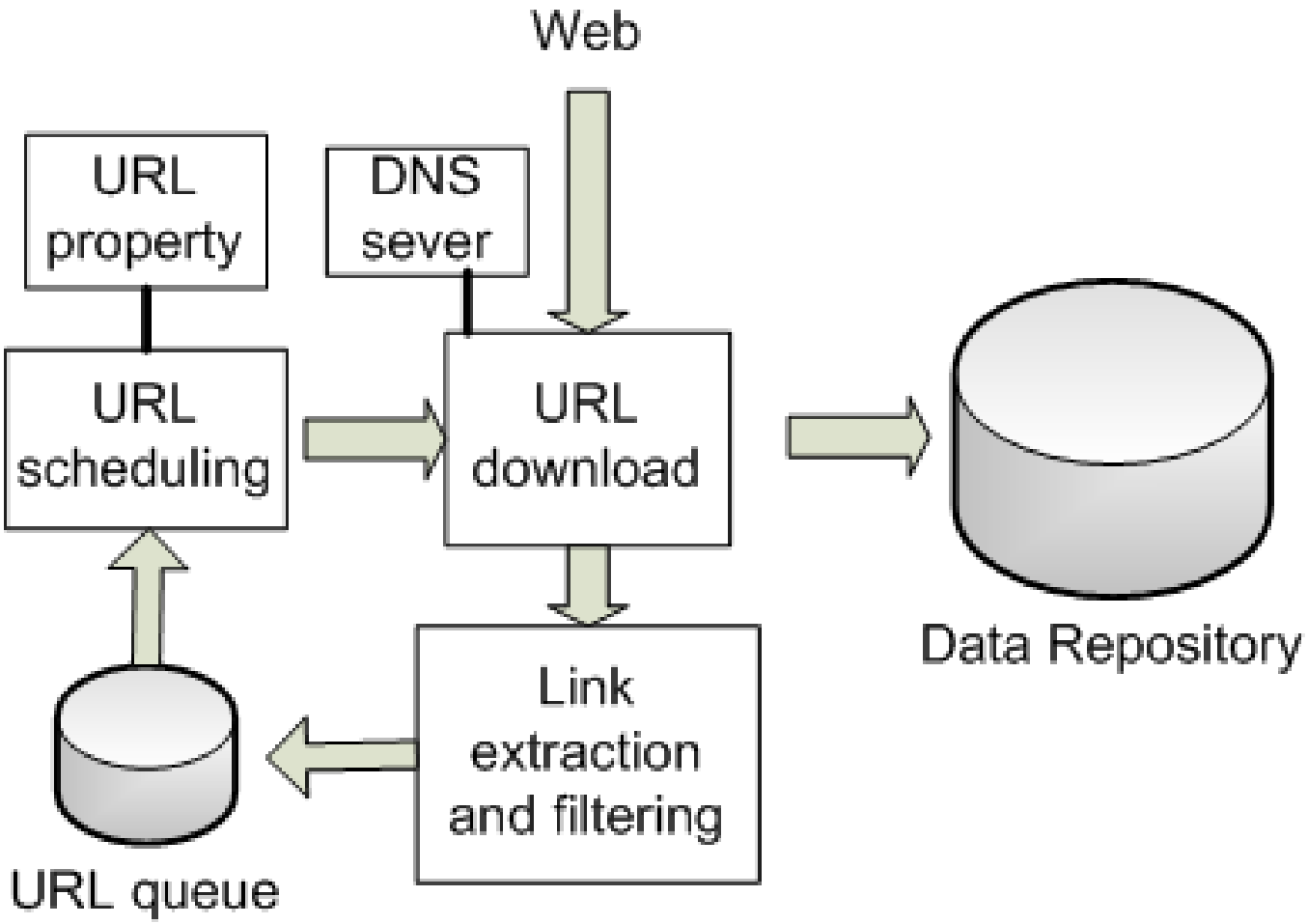
A Crawler Architecture



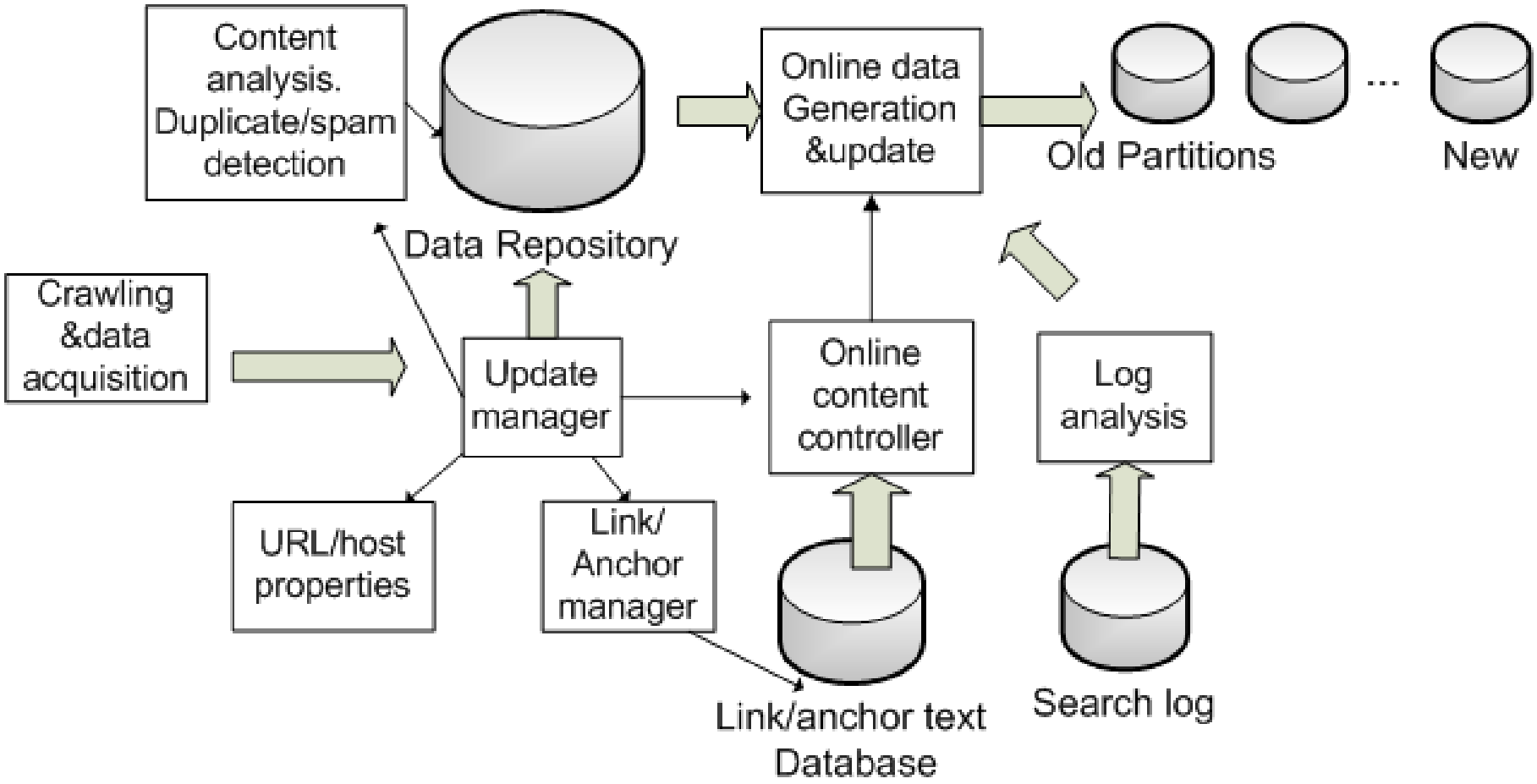
Focused Crawling

- **Attempts to download only those pages that are about a particular topic**
  - used by *vertical search* applications
  - E.g. crawl and collect technical reports and papers appeared in all computer science dept. websites
- **Rely on the fact that pages about a topic tend to have links to other pages on the same topic**
  - popular pages for a topic are typically used as seeds
- **Crawler uses *text classifier* to decide whether a page is on topic**

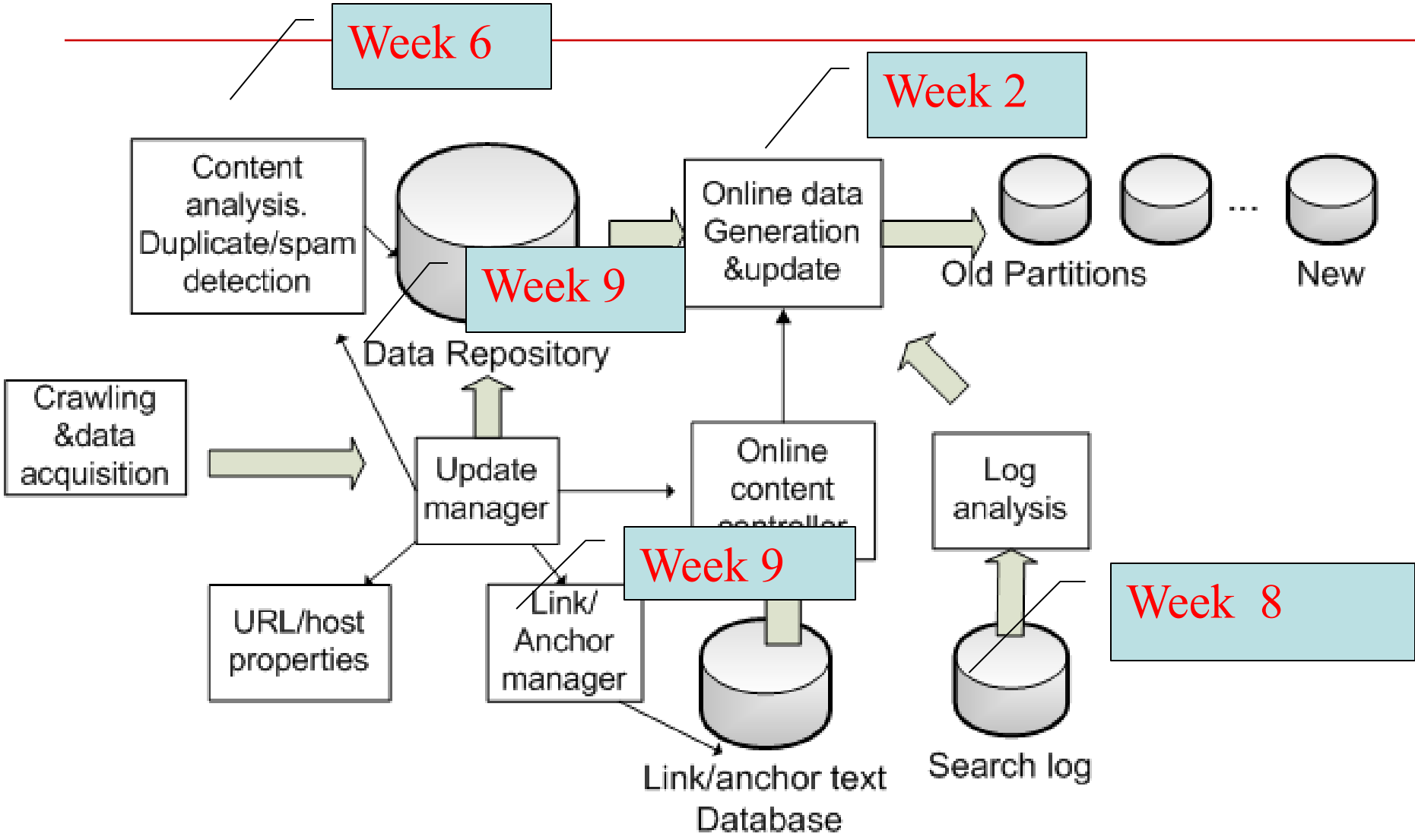
Where/what to modify in this architecture for a focused crawler?



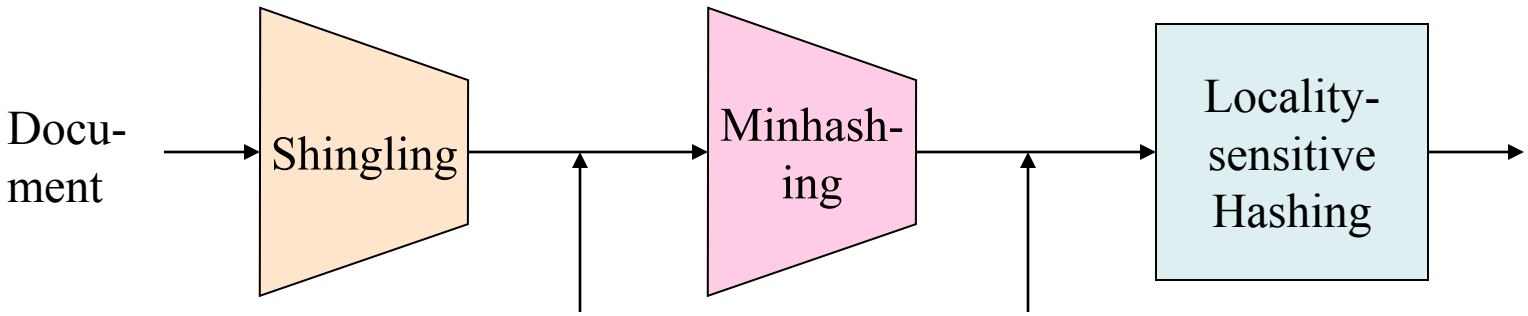
Offline Architecture at Ask



Offline Architecture at Ask



Similarity Analysis

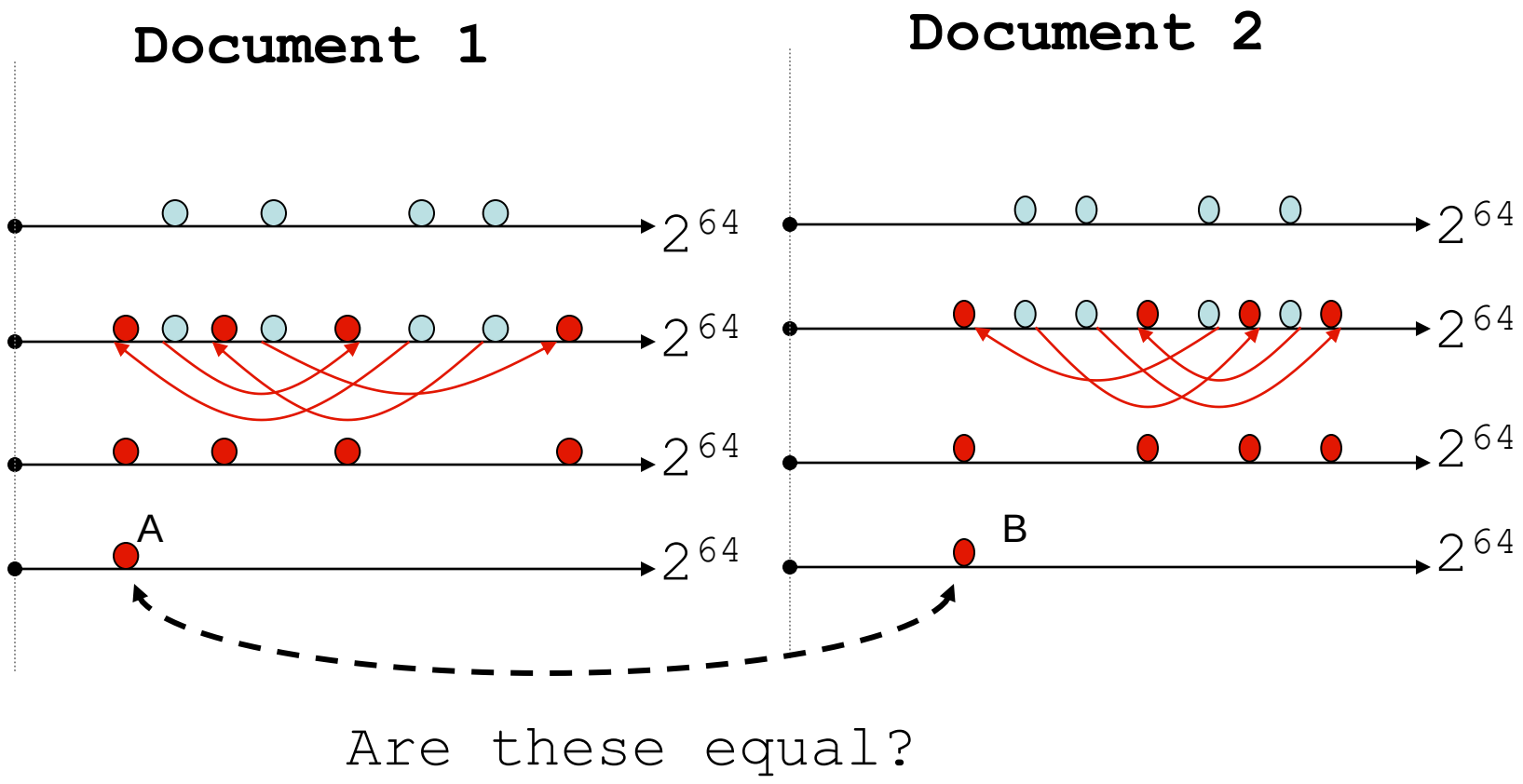


The set of strings of length *k* that appear in the document

Signatures : short integer vectors that represent the sets, and reflect their similarity

Candidate pairs : those pairs of signatures that we need to test for similarity.

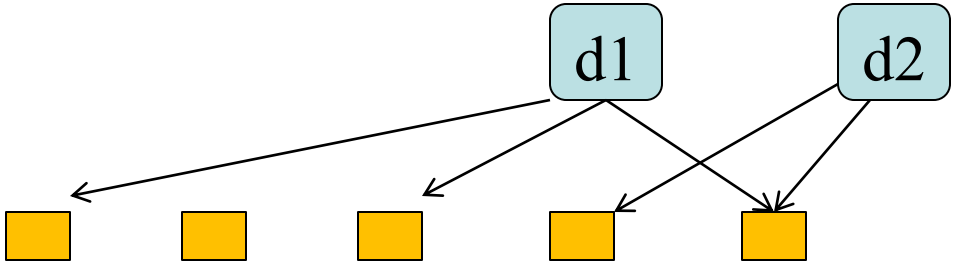
Example of Shingling and Minhash



Test for 200 random permutations: $\pi_1, \pi_2, \ldots \pi_{200}$

Locality-Sensitive Hashing

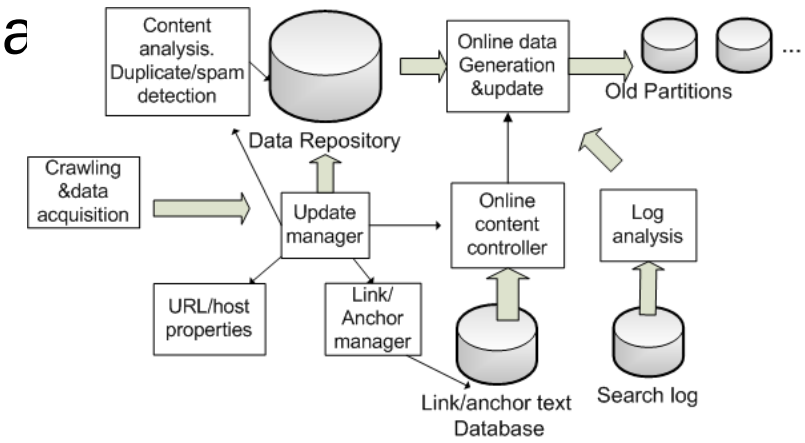
- **General idea: Use a function f(x,y) that tells whether or not *x* and *y* is a *candidate pair* : a pair of elements whose similarity must be evaluated.**
- **Map a document to many buckets**



- **Make elements of the same bucket candidate pairs.**
  - Sample probability of collision:
    - 10% similarity → 0.1%
    - 1% similarity → 0.0001%

Software Infrastructure Support at Ask.com

- **Programming support (multi-threading/exception Handling, Hadoop MapReduce)**
- **Data stores for managing billions of objects**
  - Distributed hash tables, queues etc
- **Communication and data exchange among machines/services**
- **Execution environment**
  - Controllable (stop, pause, restart).
  - Service registration and invoca
  - service monitoring
  - Logging and test framework.



Requirements for Data Repository Support in Offline Systems

- **Update**
  - handling large volumes of modified documents
  - adding new content
- **Random access**
  - request the content of a document based on its URL
- **Compression and large files**
  - reducing storage requirements and efficient access
- **Scan**
  - Scan documents for text mining.

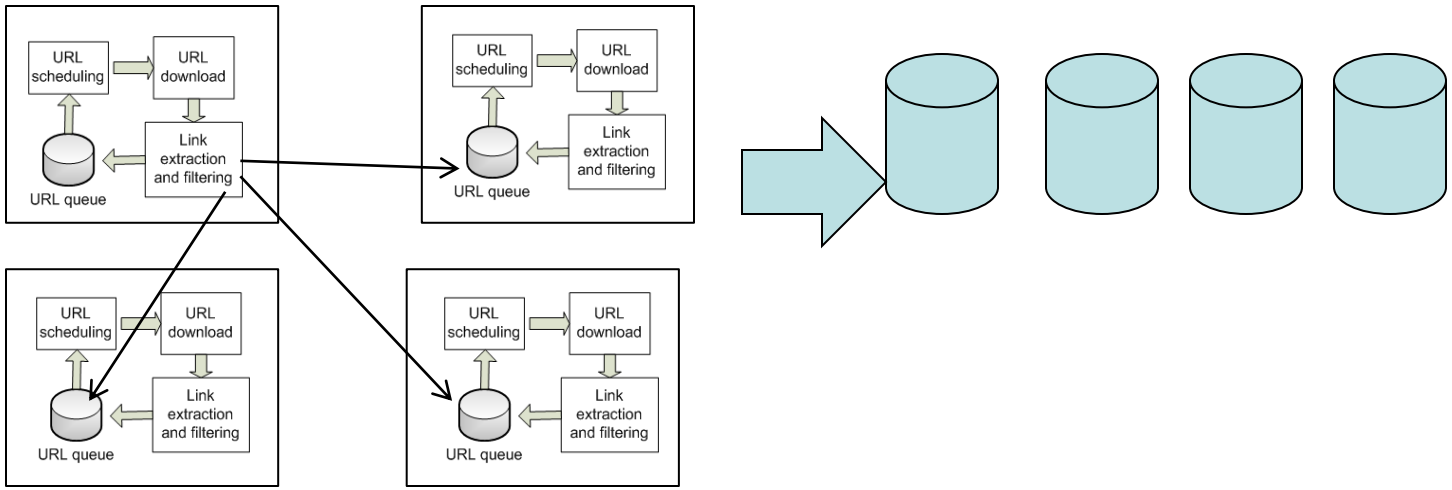
Options for Key-value Data Stores

- **Support: append or put. get operations**
- **Bigtable at Google**
- **Dynamo at Amazon**
- **Open source software**

	Technology	Language Platform	Users/ sponsors
Apache Cassandra	Bigtable Dynamo	Java/Hadoop	Apache
Hypertable	Bigtable	C++/Hadoop	Baidu
Hbase	Bigtable	Java/Hadoop	Apache
LevelDB	Bigtable	C++	Google
MongoDB		C++	

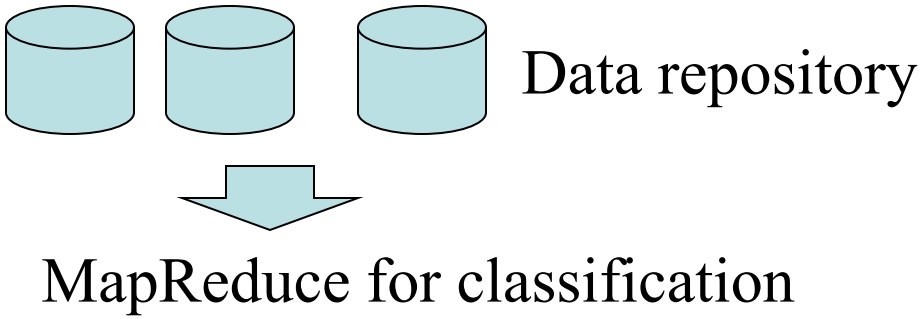
Sample Requirements for Applications: Data repository for crawling

- **Common data operations**
  - Update: Mainly append operations every day.
  - Content read:
    - Typically scan and then transfer data to another cluster
    - Sometime: random access individual pages for inspection

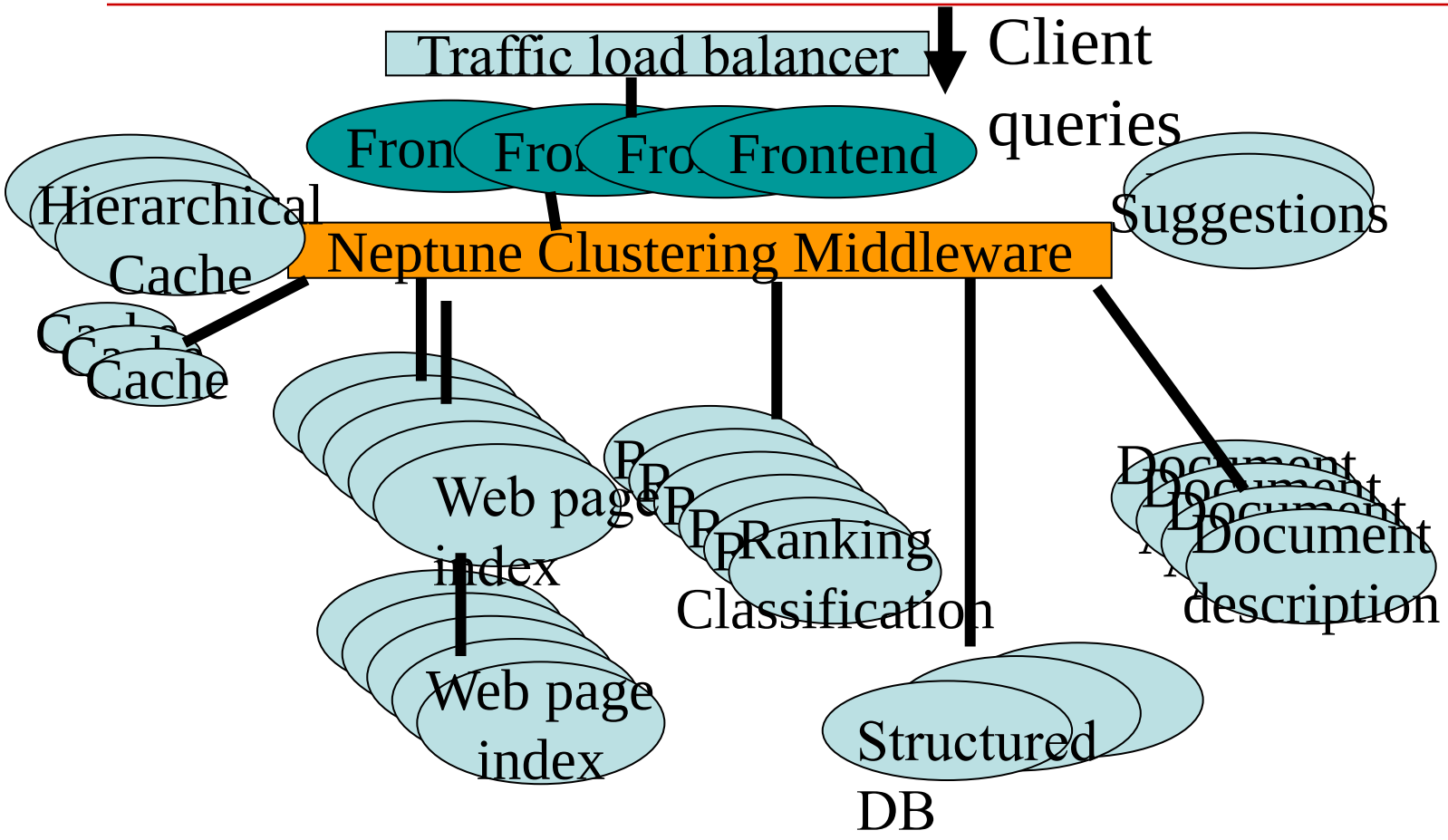


Sample Requirements for periodic data reclassification

- **Data repository hosting a large page collection with periodical page re-classification**
  - Update: Append only operations for raw data
    - Update → meta data modification periodically for selected pages (random access).
  - Read: Scan only operations for raw data processing.
    - Random read sometime for a small number of pages.

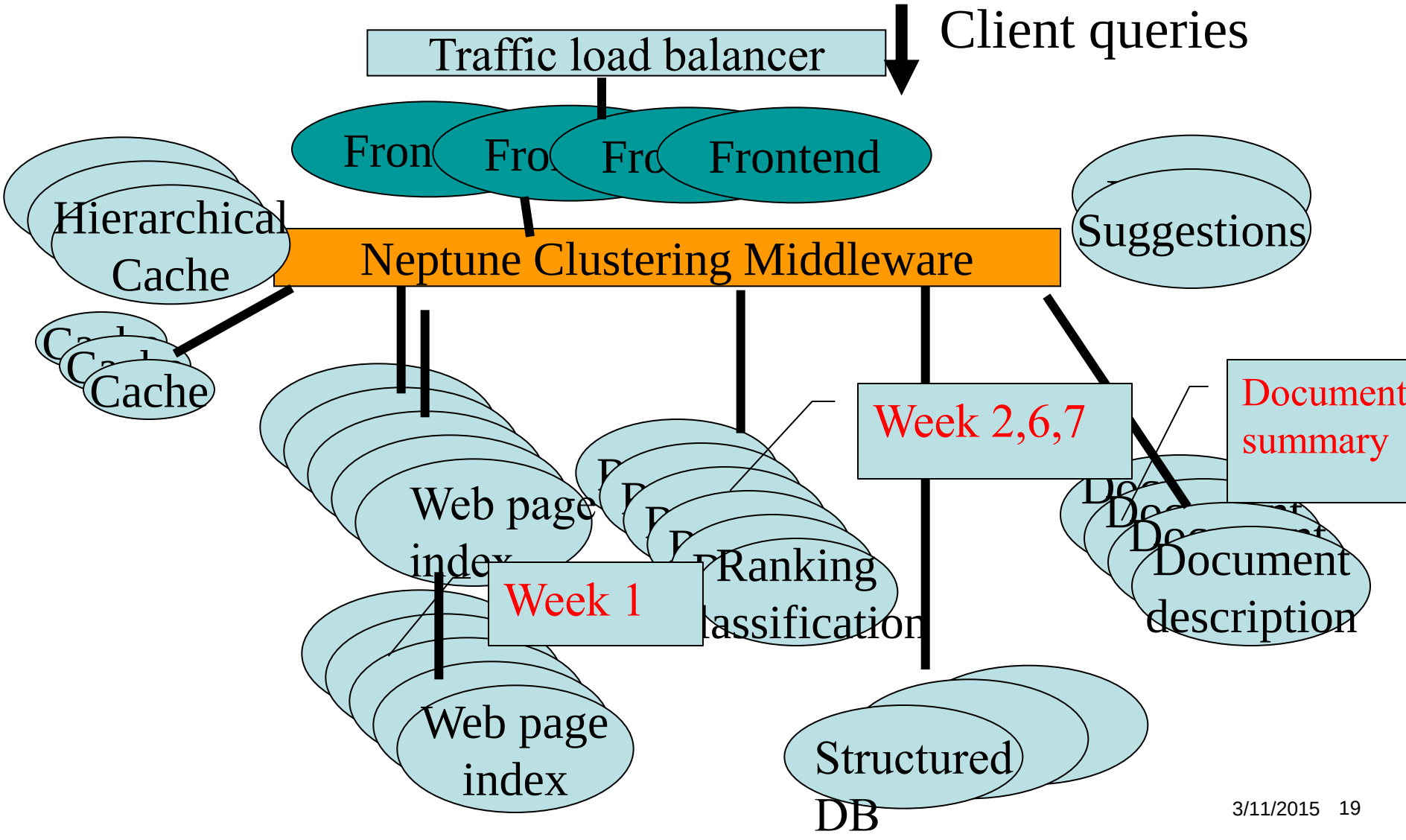


Online Engine Architecture



Web Search for a Planet: The Google Cluster Architecture
L. Barroso, J. Dean, U. Hölzle, IEEE Micro, vol. 23 (2003)

Online Engine Architecture



Document Ranking with Text, Quality, and Click Features

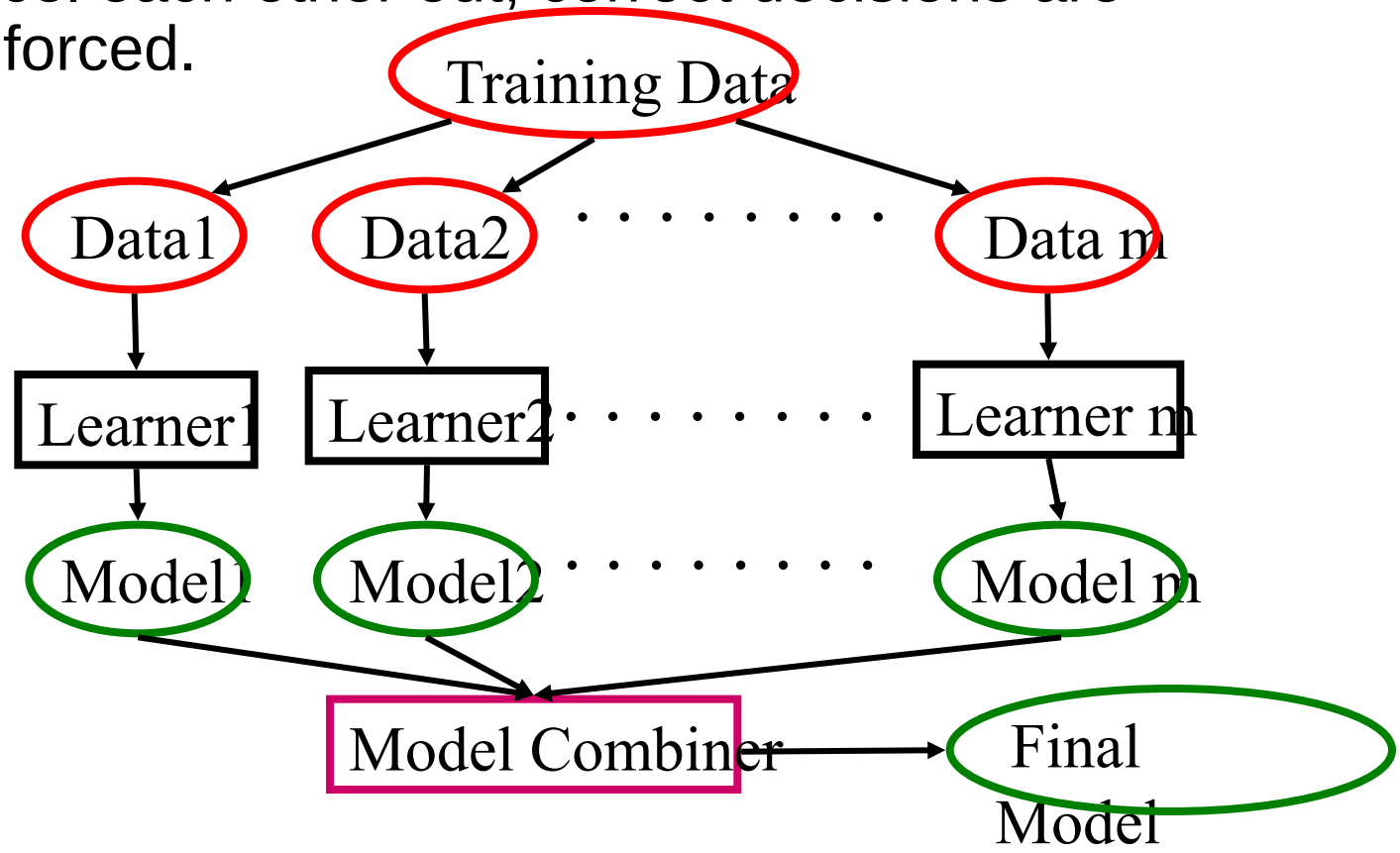
- **Text features**
  - TFIDF, BM25
  - Where do they appear? Title/body
  - Proximity (word distance)
- **Document quality and classification**
  - Web link scores (e.g. PageRank).
  - Page length, URL type etc.
- **User behavior data**
  - **Presentation**: what a user sees *before* a click
  - **Clickthrough**: frequency and timing of clicks
  - **Browsing**: what users do *after* a click

Learning to rank

- Convert ranking problem to a classification problem.
  - *Point-wise* learning
    - Given a query-document pair, predict a score (e.g. relevancy score)
  - *Pair-wise* learning
    - the input is a pair of documents for a query
  - *List-wise learning*
- Bayes, SVM, decision trees, human rules.
- Bagging/boosting to combine multiple schemes

Learning Ensembles

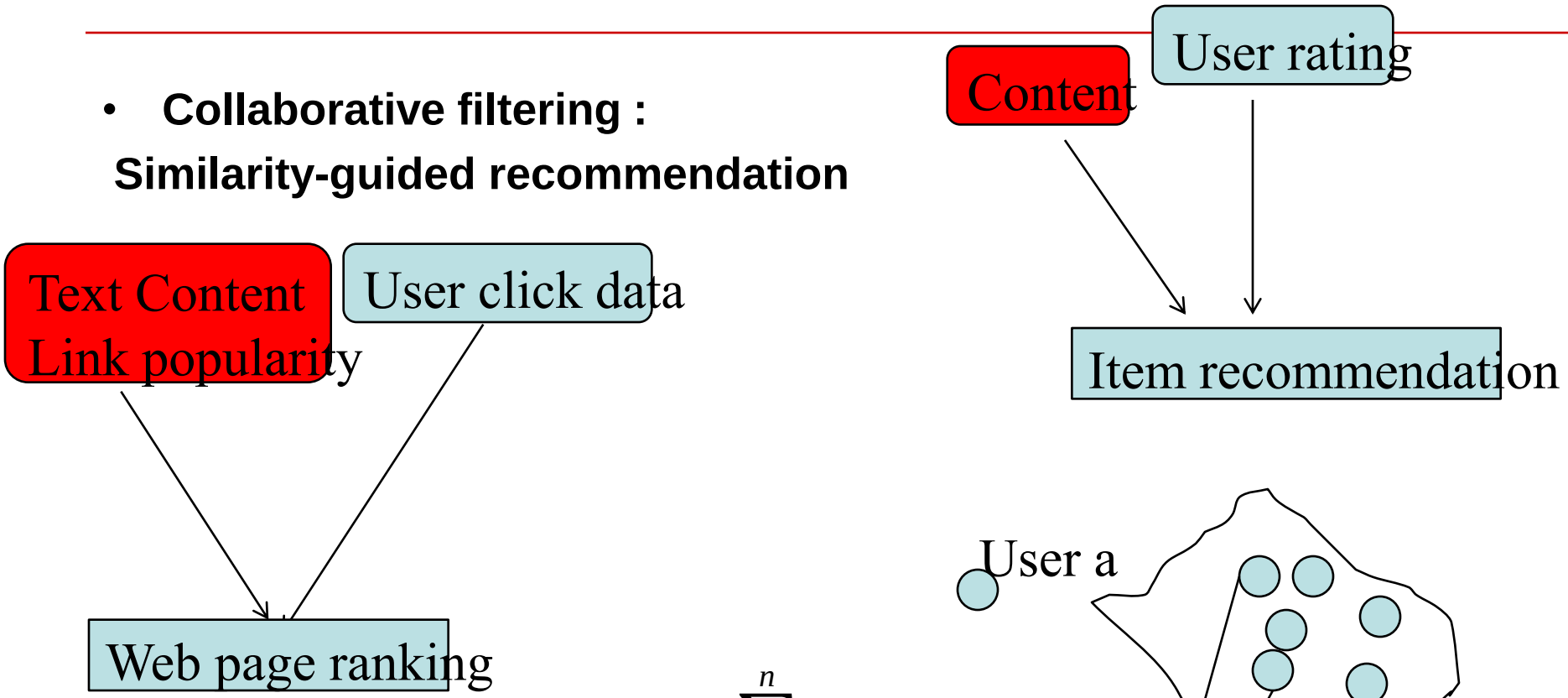
- **Learn multiple classifiers separately**
- **Combine decisions (e.g. using weighted voting)**
- When combing multiple decisions, random errors cancel each other out, correct decisions are reinforced.



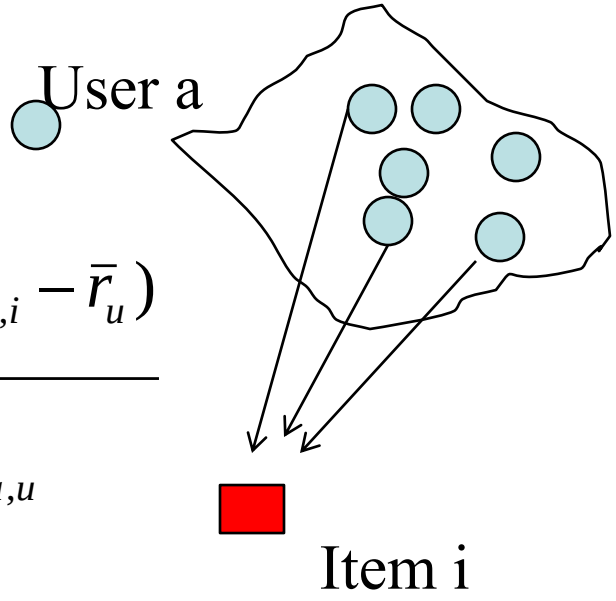
Recommendation vs Search Ranking

- **Collaborative filtering :**

  Similarity-guided recommendation

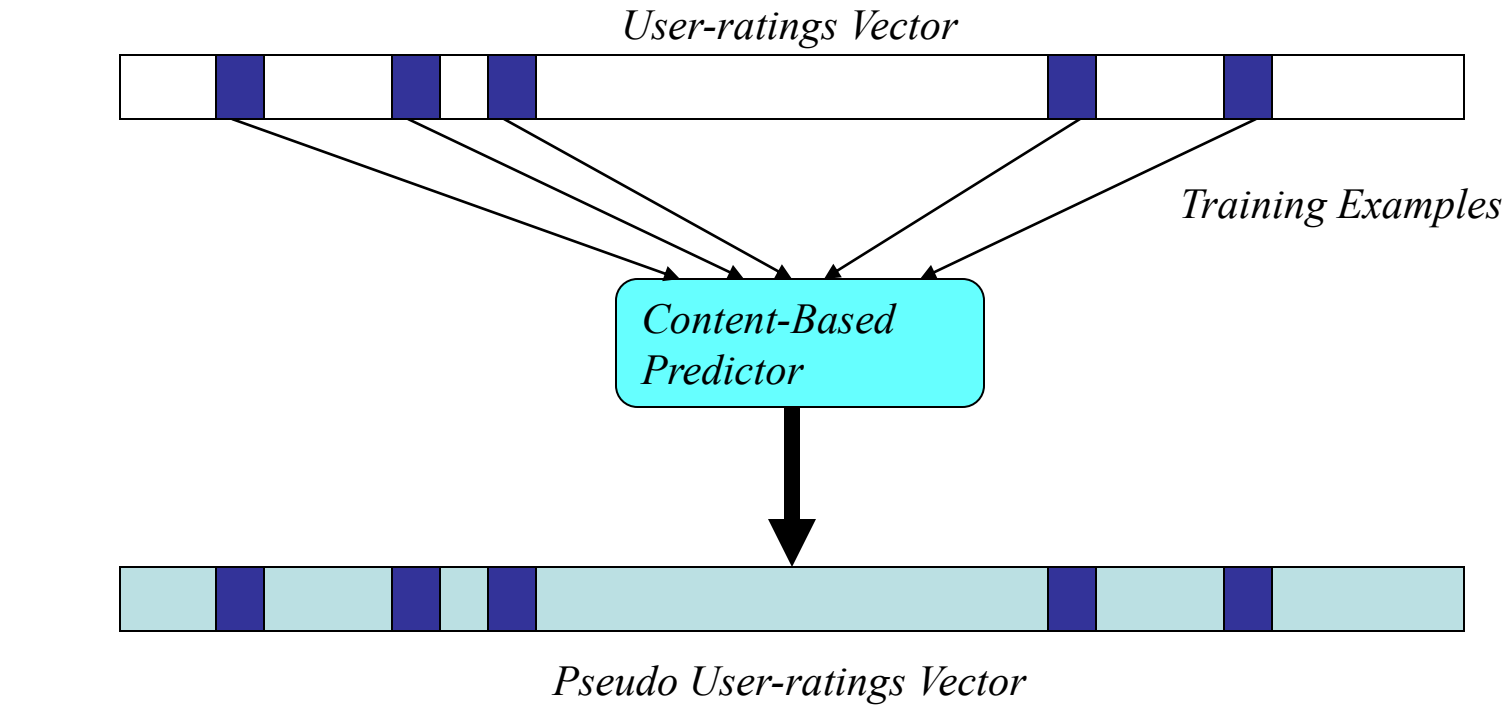


$$p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^{n} w_{a,u}(r_{u,i} - \bar{r}_u)}{\sum_{u=1}^{n} w_{a,u}}$$

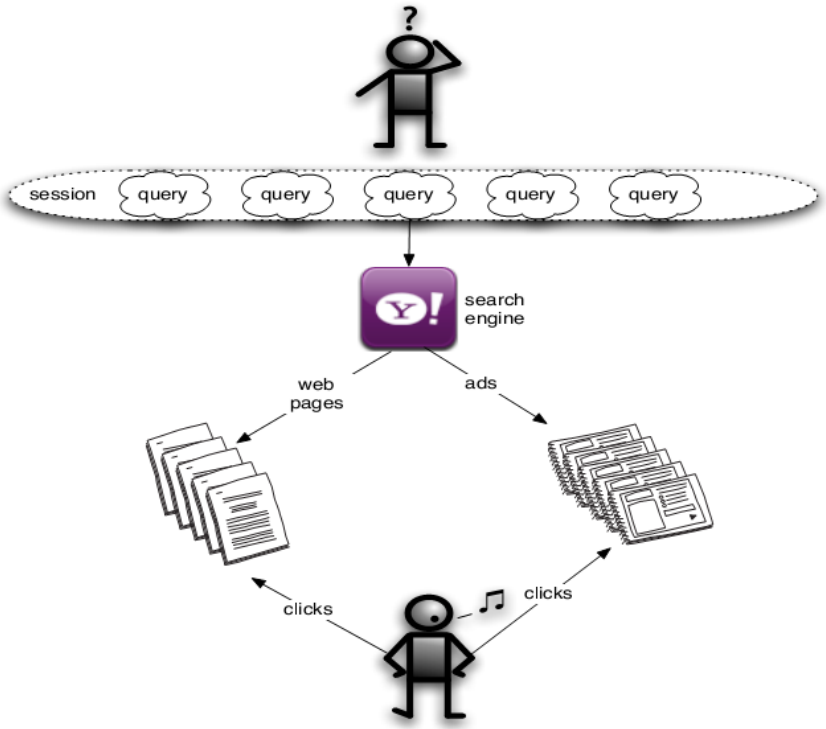


Content-Boosted Collaborative Filtering with a Sparse Rating Matrix Vector

Combine content-based prediction with user rating



Search Advertisement



Bid phrase: computational advertising
Bid: $0.5

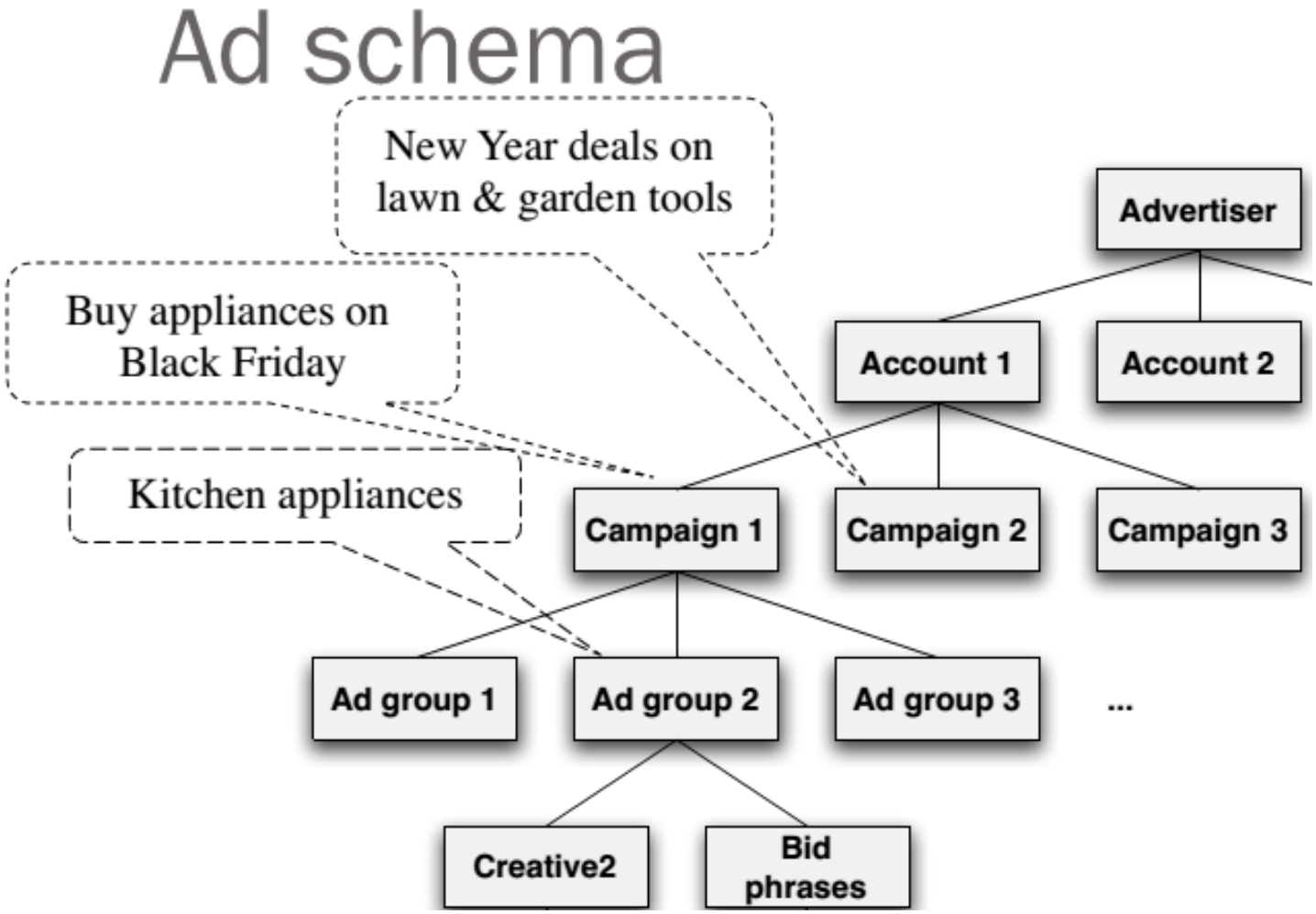
Title { ACL-08:HLT Tutorial

Creative { **Computational Advertising** Tutorial
Columbus, OH - June 15, 2008

Display URL { research.yahoo.com

Landing URL: http://research.yahoo.com/tutorials/acl08_compadv/

Search advertisement



Query-advertisement matching

- **Responsive**: satisfy directly the intent of the query
  - query: Realgood golf clubs
  - **ad**: Buy Realgood golf clubs cheap!
- **Incidental**: a user need not directly specified in the query
  - **Related**: Local golf course special
  - **Competitive**: Sureshot golf clubs
  - **Associated**: Rolex watches for golfers
  - **Spam**: Vitamins

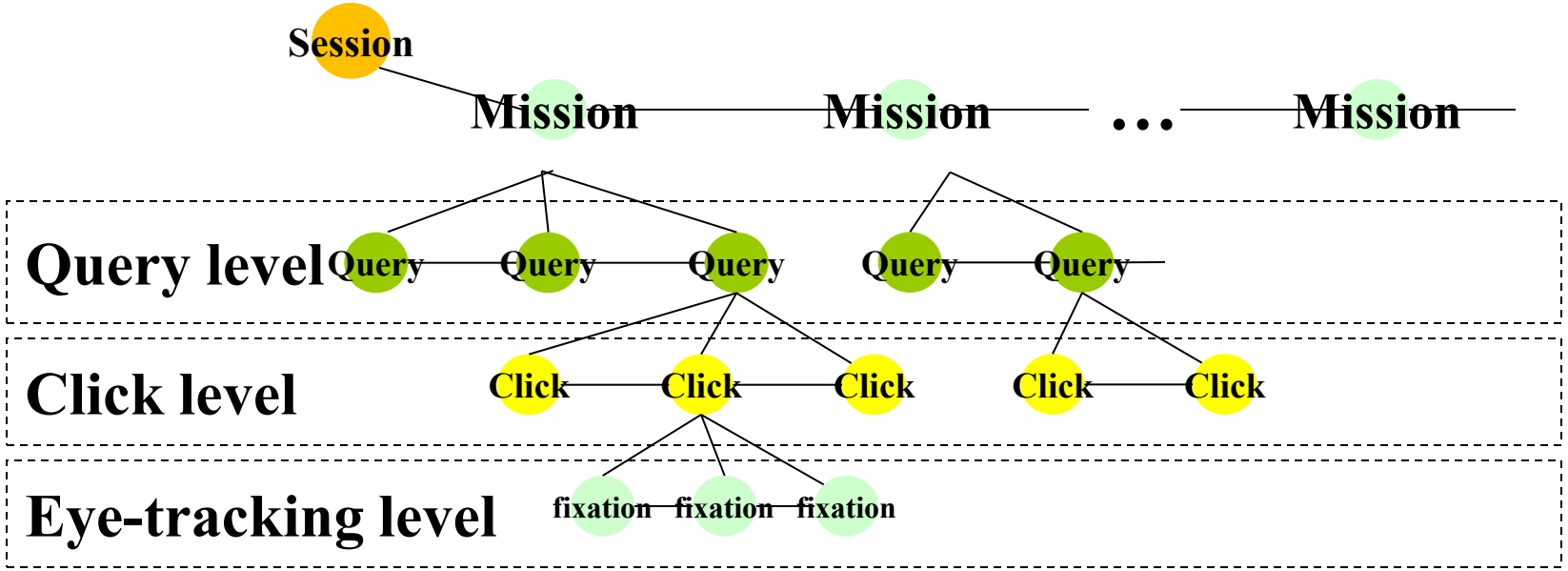
The two phases of ad selection

- **Ad Retrieval**: Consider the whole ad corpus and select a set of most viable candidates (e.g. 100)
- **Ad Reordering**: Re-score the candidates using a more elaborate scoring function to produce the final ordering
- Why do we need 2 phases:
  - Ad Retrieval:
    - considers a larger set of ads, using only a subset of available information
    - might have a different objective function (e.g. relevance) than the final function
  - Ad Reordering
    - Limited set of ads with more data and more complex calculations
    - Must use the bid in addition to the retrieval score (e.g. revenue as criteria for the ordering, implement the marketplace design()

Keyword suggestion – related problem

- Guessing the keyword for the advertiser has some risks
  - Tolerance/value of precision vs. volume differs among advertisers
  - Additional issue: what to charge the advertiser in advanced match
- Semi-automatic approach:
  - Propose rewrites to advertisers
  - Let them chose which ones are acceptable
  - Advertiser determines the bid
- Keyword suggestion tools draw upon similar data and technologies as advanced match

User Behavior Analysis with Query Sessions



Query-URL correlations:

- Query-to-pick
- Query-to-query
- Pick-to-pick

Topic Summary: Data-Driven & Large-Scale

- **Information Retrieval and Web Search**
  - Crawling, Indexing, Compression, and online retrieval/matching
  - Learning-to-rank with text/ link/click analysis.
- **Text Mining**
  - Similarity analysis. Text Categorization and Clustering. Recommendation
- **Advertisement**
- **Systems Support**
  - Online servers and offline computation.
  - Caching. MapReduce. Key-value stores. Document parsing.
  - Open source systems

T. Yang, A. Gerasoulis, Web Search Engines: Practice and Experience . *Computer Science Handbook* (T. Gonzalez. Eds), 2014. Chapman & Hall/CRC Press.