CMSC828G Principles of Data Mining Lecture #25

- Today's Reading:
  - HMS ch. 12, articles on web page
- Today's Lecture:
  - Data Organization
  - Link Analysis
- Upcoming Due Dates:
  - Project Writeup due today
  - Project Reviews due 5/7

Peer Review

- 1-2 pages
- summary
  summarize the paper succinctly and dispassionately. This is not the place to criticize. Perhaps discuss how it fits into the big picture.
- general comments
  Give the big critical picture, before sinking into the details. This is the place to take a breath, keep your perspective, and explain what the papers weaknesses are and whether they are serious, or intrinsic to our current state of knowledge, or whatever.
- constructive criticism
  not only of technical issues, but also organization and clarity.
- minor errors
  typos and grammatical errors, and minor textual problems. It's not the reviewer's job to copy edit the paper, so don't go out of your way to look for typos. If there are serious grammatical problems, please say so but please be charitable, especially if English is not the author's native language.
- from http://www.cs.unm.edu/~bap/how-to-review.html

Project Presentation

- 10 minutes
  - describe your data mining objective
  - problem definition and algorithms
  - experimental evaluation
    - methodology
    - results
  - lessons learned/advice

Data Organization

- data mining distinguished from other data analysis by the sheer **quantity** of data
- **n** number of rows (may be millions)
  - algorithms that are exponential in n are unusable
  - algorithms that are O(n) or O(n log n) are generally feasible such as:
    - counting frequencies
    - finding the mode
    - sorting
  - multiple passes over the data may not be feasible (even if algorithm is linear time)
- **p** number of columns (may be thousands)
  - for large p, even $O(p^2)$ algorithm may be infeasible

Computer Architecture 101

- Memory hierarchy

memory	*size*	*access time*	*access*
registers	100 words		direct access
on-board cache	16-1000 KB	20 ns	associative access
main memory	16 MB – several GB	70 ns	direct access
disk memory	1 – 1000s GB	10 ms	sequential access
magnetic tape	gigabytes, terabytes	seconds, minutes	sequential access

goal: increase **locality** of reference

Indexing Structures

- An **index** on an attribute A is **data structure** that makes it possible to access data points with a particular value of A more efficiently than a linear scan.

Data Structures 101

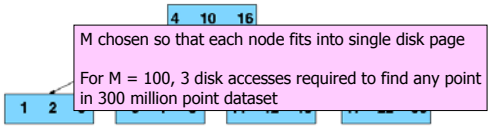
- Set of S data vectors = {x(1),…,x(n)}
- Queries:
  - point queries: want to find all points having a particular value of an ordinal attribute x.A
  - range queries: all points with $b \leq x.A \leq c$
- Simplest solution?

  binary tree
- Time complexity?

  if tree balanced, O(log n) * # of records

B*-trees

- idea similar to BST, used to access data on disk



B*-tree of degree M is tree where:
- all leaves are at same depth
- each leaf contains between M/2 and M keys
- each interior node (except possibly the root) has $M/2 \leq K \leq$ M children, with values $a_1,\ldots,a_{k-1}$
- values stored at the leaves of subtree C_i are larger than a_{i-1} and at most a_i

Hash Indices

- based on idea: If number of possible values for attribute A is small, keep list with pointer to all points with attribute value A = a.
- If number of possible values for A is large, use hash function h: Dom(A) $\rightarrow$ {1,…,M}
- r[j] is list of pointer to records x_i such that $h(a_i) = j$
- To find all points with A = a
  - for each element x_i in r[h(a)]
    - check whether $x_i.A = a$
- Typical hash function: a mod M

Multidimensional Indexing

- especially relevant in multimedia DB applications such as image databases
- example:
  - user: *retrieve all images similar to a given example image*
  - The system extracts features from the query image and searches the database for images having similar features. The features from all of the images in the DB are indexed using appropriate set of features
- types of queries: point queries, range queries, nearest neighbor queries

Types of Multidimensional Indexes

- space filling curves/z ordering/linear quadtrees The data is "linearized" and then one of the traditional index structures is used
- grid files: The data space is divided into a "grid" and indexed in each dimension.
- R-trees: Similar to B-trees where space is hierarchically partitioned and indexed at each level by region.

R-trees

- A R+-tree is a spatial data structure optimized for searching areas in N dimensional space.
- R+-tree is a spatial access method which splits space with hierarchically nested boxes. Objects are indexed in each box which intersects them. The tree is height-balanced.
- The R+-tree data structure was created by Timos Sellis, Nick Roussopoulos, and Christos Faloutsos and is described in their paper "*The R+-Tree: A Dynamic Index for Multi-Dimensional Objects*

What is Data Warehouse?

- Defined in many different ways, but not rigorously.
  - A decision support database that is maintained separately from the organization's operational database
  - Support information processing by providing a solid platform of consolidated, historical data for analysis.
- "A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision-making process."—W. H. Inmon
- Data warehousing:
  - The process of constructing and using data warehouses

Data Warehouse—Subject-Oriented

- Organized around major subjects, such as customer, product, sales.
- Focusing on the modeling and analysis of data for decision makers, not on daily operations or transaction processing.
- Provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.

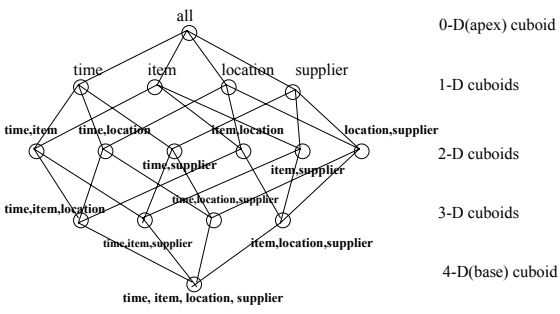
Data Warehouse vs. Operational DBMS

- OLTP (on-line transaction processing)
  - Major task of traditional relational DBMS
  - Day-to-day operations: purchasing, inventory, banking, manufacturing, payroll, registration, accounting, etc.
- OLAP (on-line analytical processing)
  - Major task of data warehouse system
  - Data analysis and decision making
- Distinct features (OLTP vs. OLAP):
  - User and system orientation: customer vs. market
  - Data contents: current, detailed vs. historical, consolidated
  - Database design: ER + application vs. star + subject
  - View: current, local vs. evolutionary, integrated
  - Access patterns: update vs. read-only but complex queries

From Tables and Spreadsheets to Data Cubes

- A data warehouse is based on a multidimensional data model which views data in the form of a data cube
- A data cube, such as sales, allows data to be modeled and viewed in multiple dimensions
  - Dimension tables, such as item (item_name, brand, type), or time(day, week, month, quarter, year)
  - Fact table contains measures (such as dollars_sold) and keys to each of the related dimension tables
- In data warehousing literature, an n-D base cube is called a base cuboid. The top most 0-D cuboid, which holds the highest-level of summarization, is called the apex cuboid. The lattice of cuboids forms a data cube.

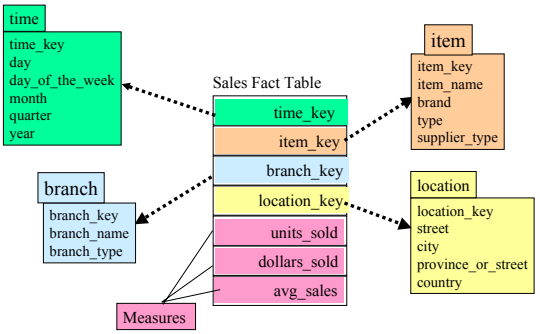
Cube: A Lattice of Cuboids



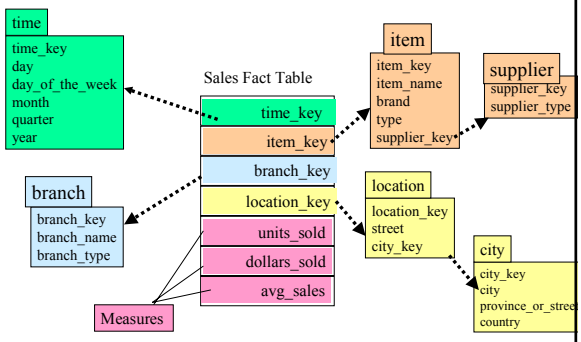
Conceptual Modeling of Data Warehouses

- Modeling data warehouses: dimensions & measures
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - Fact constellations: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

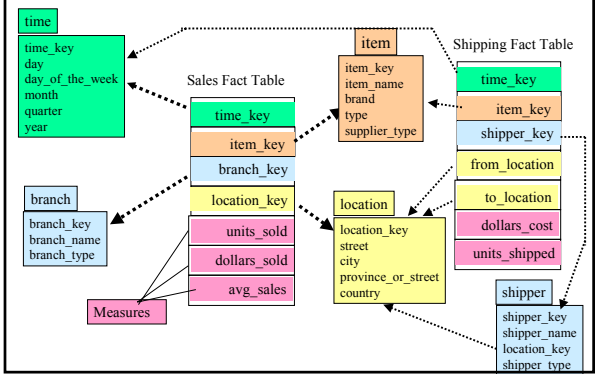
Example of Star Schema



Example of Snowflake Schema

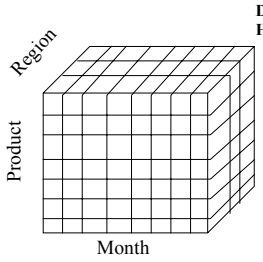


Example of Fact Constellation

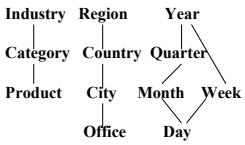


Multidimensional Data

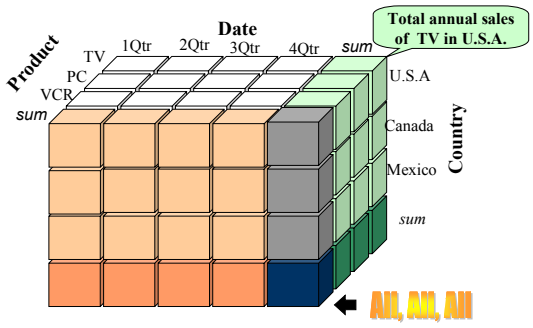
- Sales volume as a function of product, month, and region



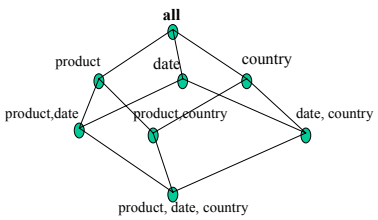
Dimensions: Product, Location, Time
Hierarchical summarization paths



A Sample Data Cube



Cuboids Corresponding to the Cube



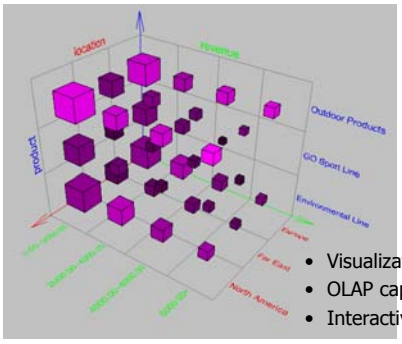
0-D(apex) cuboid

1-D cuboids

2-D cuboids

3-D(base) cuboid

Browsing a Data Cube



- Visualization
- OLAP capabilities
- Interactive manipulation

Typical OLAP Operations

- Roll up (drill-up): summarize data
  - *by climbing up hierarchy or by dimension reduction*
- Drill down (roll down): reverse of roll-up
  - *from higher level summary to lower level summary or detailed data, or introducing new dimensions*
- Slice and dice:
  - *project and select*

Data Management and Data Mining

- How to deal with massive datasets?
  - force the data into main memory
  - scalable version of data mining algorithms
  - special purpose algorithms for disk access
  - pseudo data sets and sufficient statistics

Link Analysis

Link Analysis

- Finding patterns in graphs
  - Bibliometrics – finding patterns in citation graphs
  - Sociometry – finding patterns in social networks
  - Collaborative Filtering – finding patterns in rank(person, item) graph
  - Webometrics – finding patterns in web page links

Web Link Analysis

- Used for
  - ordering documents matching a user query: ranking
  - deciding what pages to add to a collection: crawling
  - page categorization
  - finding related pages
  - finding duplicated web sites

Web as Graph

- Link graph:
  - node for each page
  - directed edge (u,v) if page u contains a hyperlink to page v
- Co-citation graph
  - node for each page
  - *undirected* edge (u,v) iff exists a third page w linking to both u and v
- assumption:
  - link from page A to page B is a recommendation of page B by A
  - If A and B are connected by a link, there is a higher probability that they are on the same topic

Connectivity-Based Ranking

- Query-independent: gives an intrinsic quality score to a page
- Approach #1: larger number of hyperlinks pointing to a page, the better the page
  - drawback?
  - each link is equally important
- Approach #2: weight each hyperlink proportionally to the quality of the page containing the hyperlink

Page Rank

- PageRank R(p) of page p:

$$R(p) = \varepsilon / n + (1-\varepsilon) \cdot \sum_{(q,p)\in G} \frac{R(q)}{\text{out degree}(q)}$$

- where
  - ε is a dampening factor usually set between 0.1 and 0.2
  - n is the number of nodes in G
  - outdegree(q) is the number of edges leaving page p

Alternate Formulation: Random Surfer

- The random surfer can follow any outlink from a page with equal probability. Periodically the random surfer gets bored and jumps to a random page on the Web.
- Page rank is the stationary distribution of infinite walk p_1, p_2,p_3, …
  - Each node is equally likely to be the starting node
  - At node p_i
    - with probability ε, node p_{i+1} is chosen uniformly at random from all the nodes in G
    - with probability 1-ε, node p_{i+1} is chosen uniformly at random from the nodes q in G s.t. (p_{i+1}, q)

PageRank cont.

- PageRank is the dominant eigenvector of the probability transition matrix of the random walk
- When PageRank is computed iteratively using the previous equation, the computation will eventually converge (under some weak assumptions on the values in the probability matrix)
- Typically 100 iterations suffice to converge.
- Advantage: not query specific, can be done once

Query-dependent Connectivity-Based Ranking

- Carrier and Kazman
- For each query, build a subgraph of the link graph G limited to pages on query topic
- Build the neighborhood graph
  1. A start set S of documents matching query given by search engine (~200)
  2. Set augmented by its neighborhood, the set of documents that either point to or are pointed to by documents in S (limit to ~50)
  3. Then rank based on indegree

problem?

Idea

- We desire pages that are **relevant** (in the neighborhood graph) and **authoritative**
- As in page rank, not only the in-degree of a page p, but the quality of the pages that point to p. If more important pages point to p, that means p is more authoritative
- Key idea: Good **hub** pages have links to good **authority** pages
- given user query, compute a hub score and an authority score for each document
- high authority score $\rightarrow$ relevant content
- high hub score $\rightarrow$ links to documents with relevant content

HITS algorithm

- Kleinberg, 1998
- Hypertext Internet Topic Search

Kleinberg's Algorithm

1. Initialize: $h_p=1$, $a_p=1$ for all pages in neighborhood graph
2. While the vectors H and A have not converged
3. $A[i] = \sum_{(j,i)\in N} H[j]$
4. $H[i] = \sum_{(i,j)\in N} A[j]$
5. Normalize the H and A vectors

Does the algorithm converge?

- A is adjacency matrix of neighborhood graph
- At the ith iteration:
  $a_i=A^t h_{i-1}$
  $h_i=A^t a_i$
  normalize a_i and h_i
- Under mild conditions converges, convergence rate σ_1^2/σ_1^2
- $h^{(k)}$ converges to the leading left singular vector of A
- $a^{(k)}$ converges to the leading right singular vector of A

Problems

- Considers only a small part of web graph, adding a few edges can potentially change scores considerably, thus easier to manipulate scores
- If neighborhood graph contains more pages on a topic different from the query, then the top authority and hub pages are on this different topic. Called topic drift.

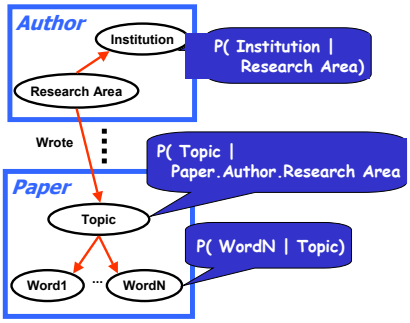
Improvements to Basic Algorithm

- Put weights on edges to reflect importance of links, e.g., put higher weight if anchor text associated with the link is relevant to query
- Normalize weights outgoing from a single source or coming into a single sink. This alleviates spamming of query results
- Eliminate edges between same domain

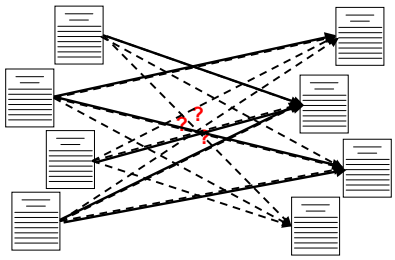
Using Link Structure for Web page classification

- Chakrabarti, et al. SIGMOD 1998.
- Mitchell and Slatterly.

Probabilistic Relational Models



Link Uncertainty

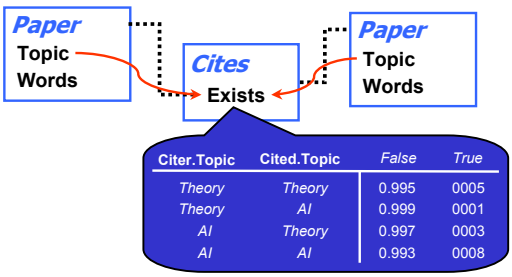


Document Collection Document Collection

PRM w/ Exists Uncertainty

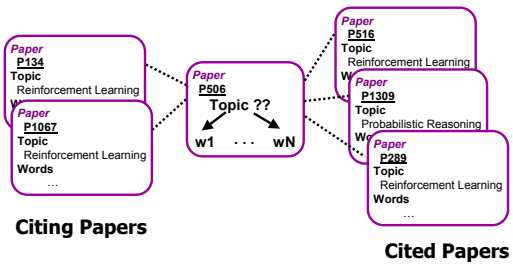
Dependency model for existence of relationship

Exists Uncertainty Example



Citer.Topic	Cited.Topic	False	True
Theory	Theory	0.995	0005
Theory	AI	0.999	0001
AI	Theory	0.997	0003
AI	AI	0.993	0008

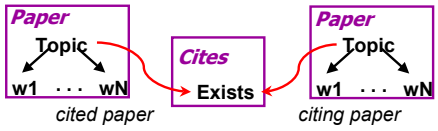
Task: Predict Topic/Category



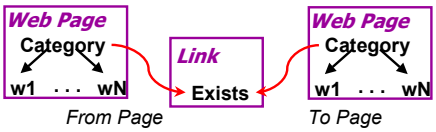
Citing Papers

Cited Papers

Domains

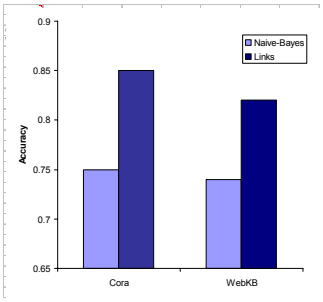


Cora Dataset, McCallum, et. al

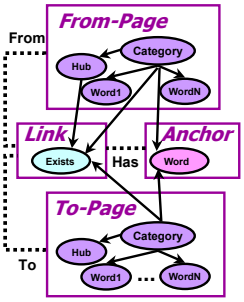


WebKB, Craven, et. al

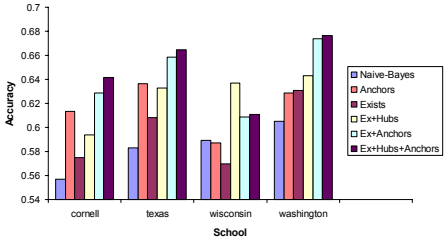
Prediction Accuracy



Web Domain



WebKB Results



Graph structure in the Web

- Understanding the various properties of the web graph – diameter, degree distributions, connected components – useful for:
  - designing better crawl strategies on the web
  - analyzing behavior of web algorithms that use link information
  - prediction of web structures, such as bipartite cores and better algorithms to compute them
  - predict emergence of new, yet unexploited, phenomena in the web graph

Example Web Graph Properties

- Six degrees of separation between any 2 web pages
  - almost true in the strongly connected component part of web graph if allow traversal of both out-links and in-links
  - not true in general. small-world network models for graphs
- probability that a web page has in-degree i is proportional to $1/i^x$. Latest estimate for x=2.1
- Average path length between two web pages is 19 (Barabasi). Disputed by (Raghavan, et. al).

References

- *Principles of Data Mining*, Hand, Mannila, Smyth. MIT Press, 2001.
- Notes from Dr. M.V. Ramakrishna http://goanna.cs.rmit.edu.au/~rama/cs442/info.html
- Notes from CS 395T: Large-Scale Data Mining, Inderjit Dhillon http://www.cs.utexas.edu/users/inderjit/courses/dm2000.html
- Link Analysis in Web Information Retreival, Monika Henzinger. Bulletin of the IEEE computer Society Technical Committee on Data Engineering, 2000. research.microsoft.com/research/db/debull/A00sept/henzinge.ps
- slides from *Data Mining: Concepts and Techniques*, Jan and Kamber, Morgan Kaufman, 2001.