CS294-1 Behavioral Data Mining

Prof: John Canny
GSI: Kenghao Chang

1997

BackRub Search: university

university Search

BackRub Query Results

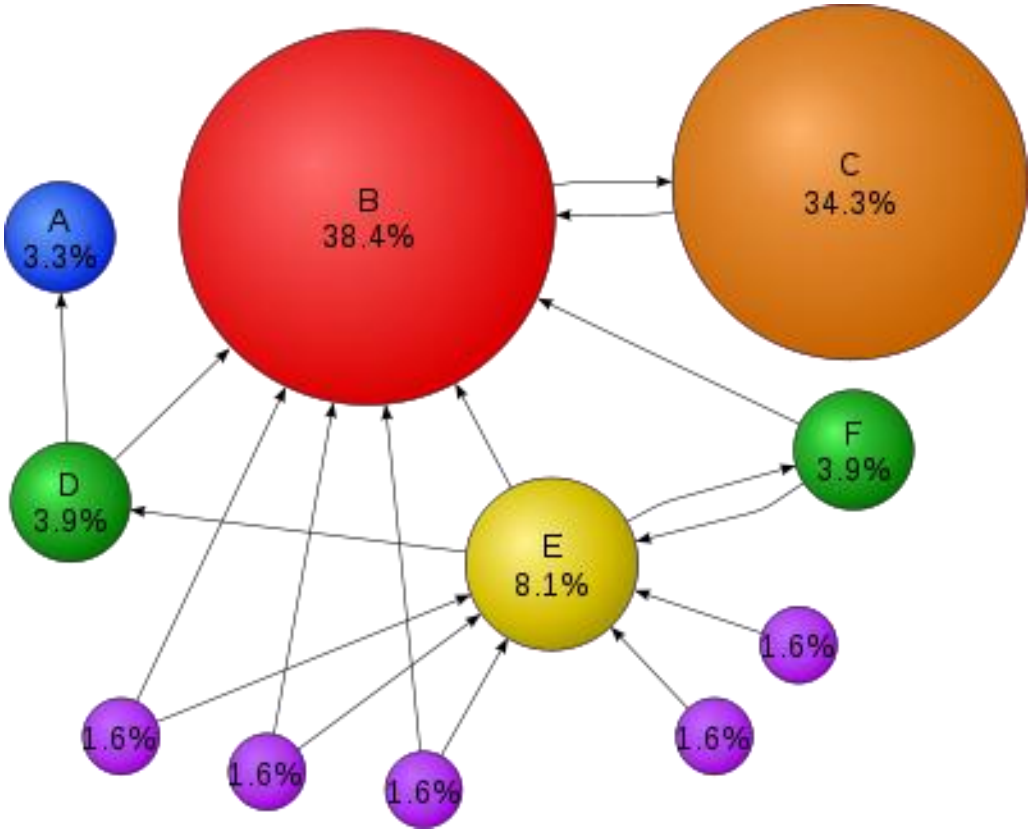
BackRub's Highest Ranked Sites

University of Illinois at Urbana-Champaign
http://www.uiuc.edu/
694.687 8460 backlinks *12k - 10/25/96 - 11/1/96*

Stanford University Homepage
http://www.stanford.edu/
609.303 8857 backlinks *4k - none - 11/1/96*

Stanford University: Portfolio Collection
http://www.stanford.edu/home/administration/portfolio.html
167.919 34 backlinks

Pagerank: The web as a behavioral dataset

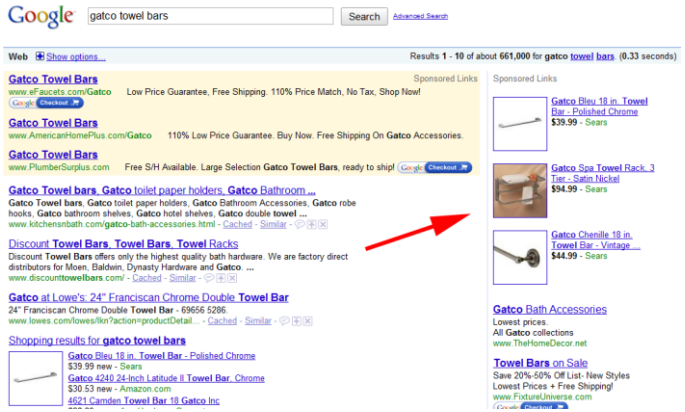


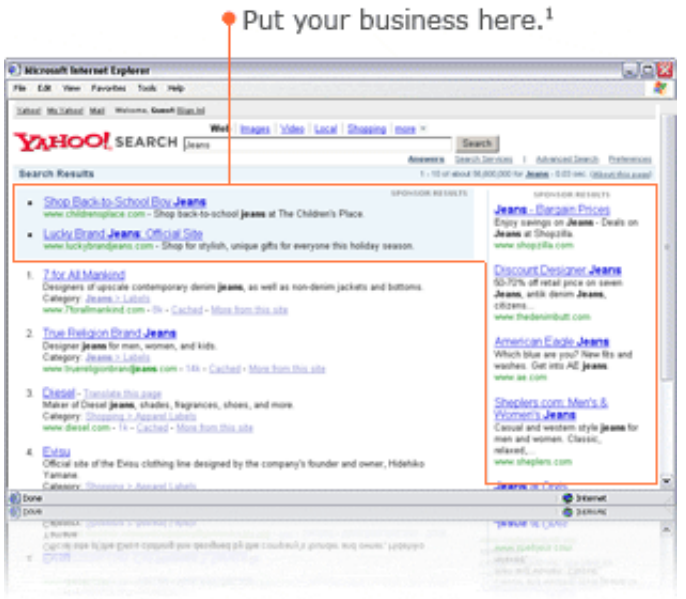
DB size =1-2 trillion sites

Google server farms
2 million machines (est ?)

1998 – sponsored search

Overture



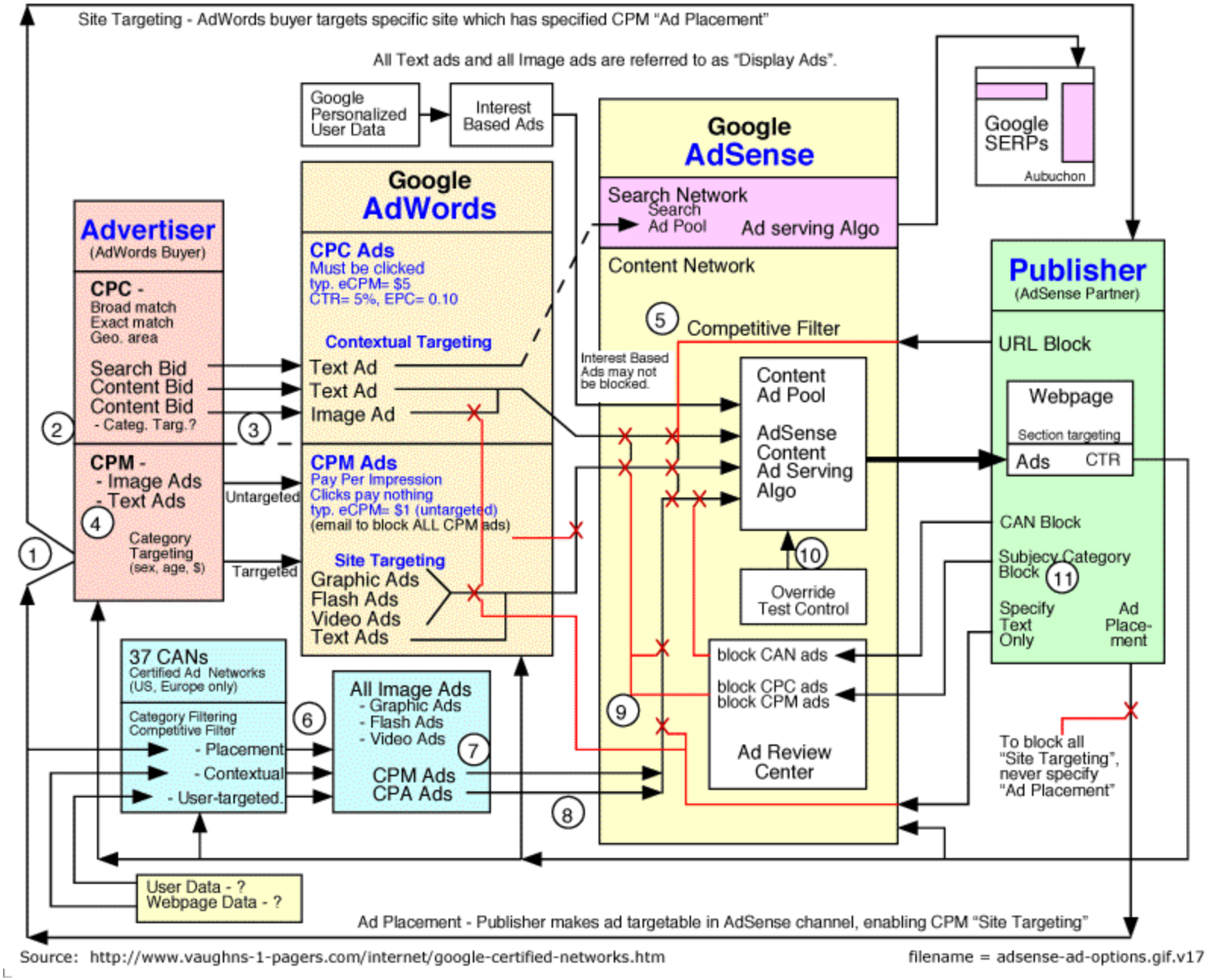


2002

Sponsored search

- Google revenue around $28 bn last year from marketing, 97% of the companies revenue.
- Sponsored search uses an auction – a pure competition for marketers trying to win access to consumers.
- In other words, a competition for **models** of consumers – their likelihood of responding to the ad – and of determining the right bid for the item.
- There are around 20 billion search requests a month. Perhaps a **trillion events** of history between search providers.
- Features that matter: source URL, time-of-day, geo, demographics, and user history. Increasingly real-time.

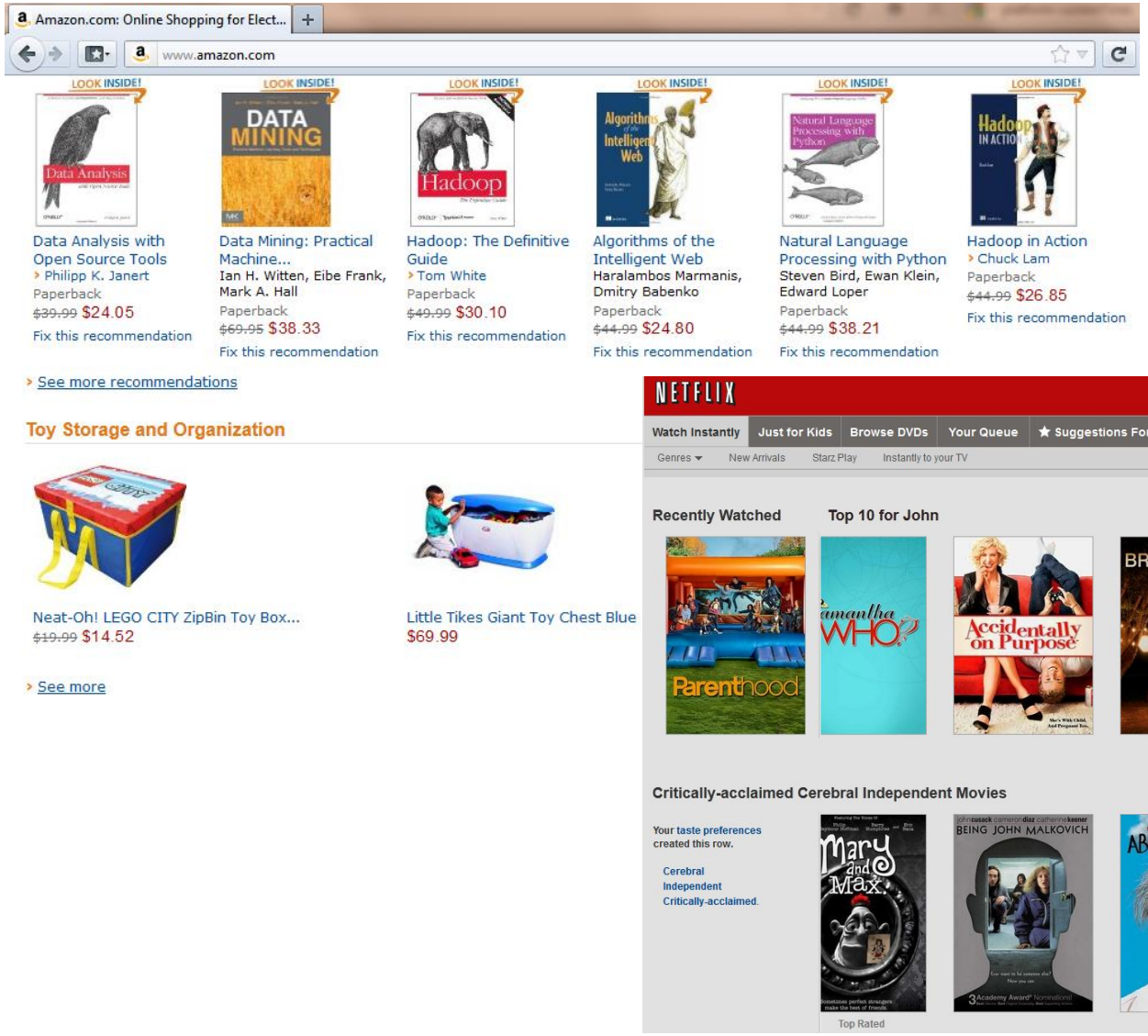

TOP 20
Keyword Categories
Percentages correspond to the number of keywords in the top 10,000 keywords that belong to that category.
#1
$54.91
Top CPC
Insurance
24%
auto insurance price quotes
Search
ca automobile insurance
building contents insurance
buy car insurance online
life insurance comparison quotes
Top Sample Queries
#2
Loans
$44.28
Top CPC
consolodate graduate student loans
Search
fixed home equity loan rates
cheapest homeowner loans
fixed rate secured loans
Top Sample Queries
12.8%
#3
Mortgage
9%
$47.12
Top CPC
#4
Attorney
$47.07
Top CPC
3.6%
#5
3.2%
3%
2.5%



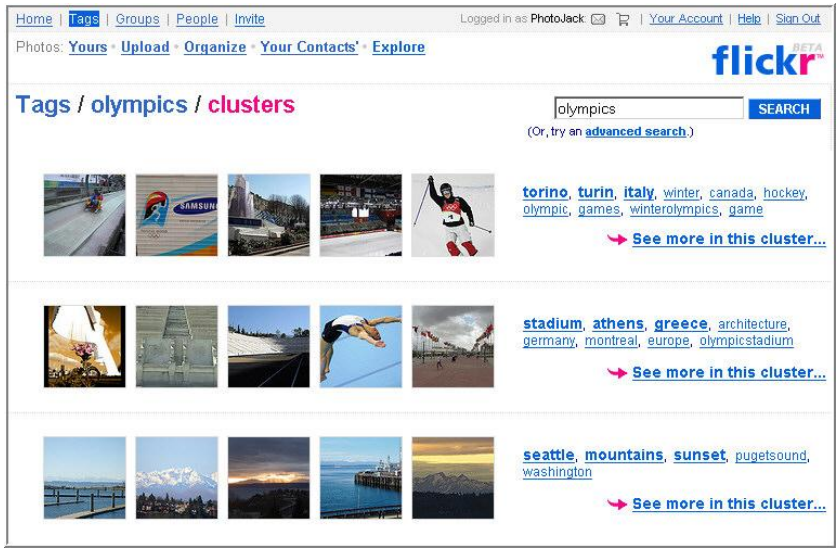
Source: http://www.vaughns-1-pagers.com/internet/google-certified-networks.htm

filename = adsense-ad-options.gif.v17

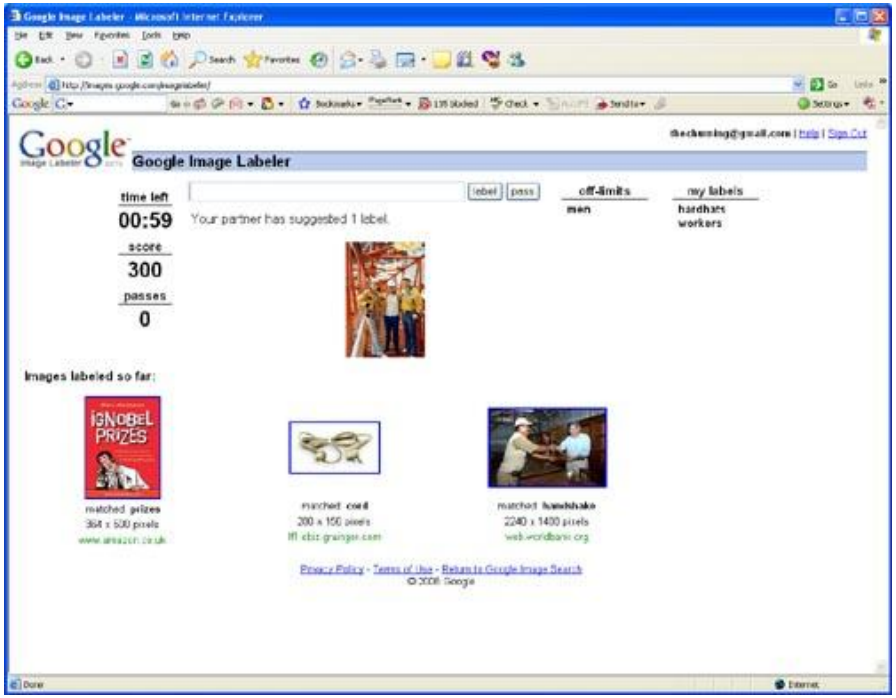
Recommenders



Tag data + Implicit tagging



Flickr hosts around 6 billion images



Based on Luis Von Ahn's ESP game

Pulse of the Nation:

U.S. Mood Throughout the Day, as inferred from Twitter

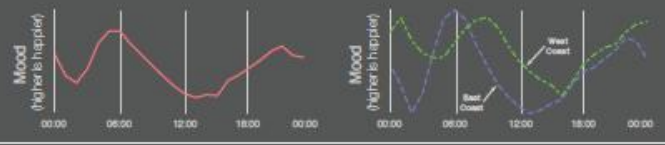
All times are Eastern Standard Time (EST)

Happier

00:00 01:00 02:00 03:00 04:00 05:00 06:00 07:00 08:00 09:00 10:00 11:00 12:00 13:00 14:00 15:00 16:00 17:00 18:00 19:00 20:00 21:00 22:00 23:00

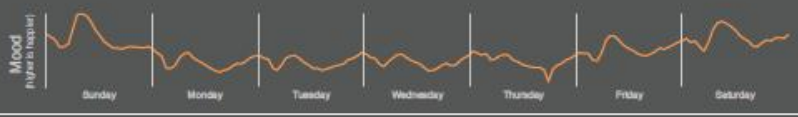
Mood Variations

A number of interesting trends can be observed in the data. First, overall daily variations can be seen (first graph), with the early morning and late evening having the highest level of happiness. Second, geographic variations can be observed (second graph), with a significantly happier west coast that is consistently three hours behind the east coast.



Weekly Variations

Weekly trends can be observed as well, with weekends much happier than weekdays. The peak in mood is on Sunday mornings, and the trough occurs on Thursday evenings.

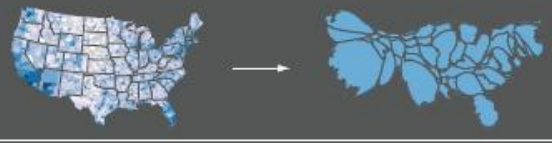


About the Data and Visualization

The plots were calculated using over 300 million tweets (Sep 2006 – Aug 2009) collected by MPI-SWS researchers, represented as density-preserving cartograms. The mood of each tweet was inferred using ANEW word list (Bradley, M.M., & Lang, P.J. *Affective norms for English words (ANEW): Stimuli, instruction manual and affective ratings*. Technical report C-1, The Center for Research in Psychophysiology, University of Florida). County area data was taken from the U.S. Census Bureau at http://factfinder.census.gov, and the base U.S. map was taken from Wikimedia Commons. User locations were inferred using the Google Maps API, and mapped into counties using PostGIS and U.S. county maps from the U.S. National Atlas. Mood colors were selected using Color Brewer 2.

About Cartograms

A cartogram is a map in which the mapping variable (in this case, the number of tweets) is substituted for the true land area. Thus, the geometry of the actual map is altered so that the shape of each region is maintained as much as possible, but the area is scaled in order to be proportional to the number of tweets that originate in that region. The result is a density-equalizing map. The cartograms in this work were generated using the cart software by Mark E. J. Newman.

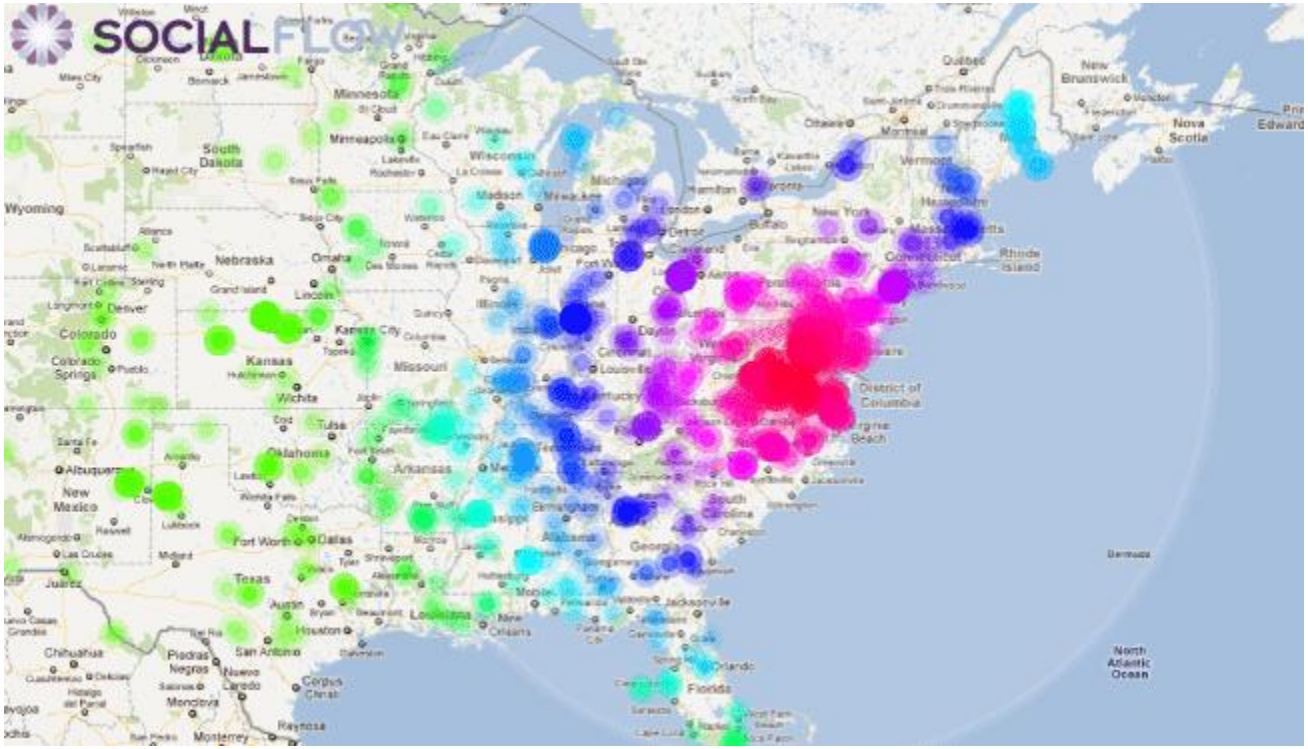


Northeastern University
College *of* Computer and Information Science †
Center for Complex Network Research ‡

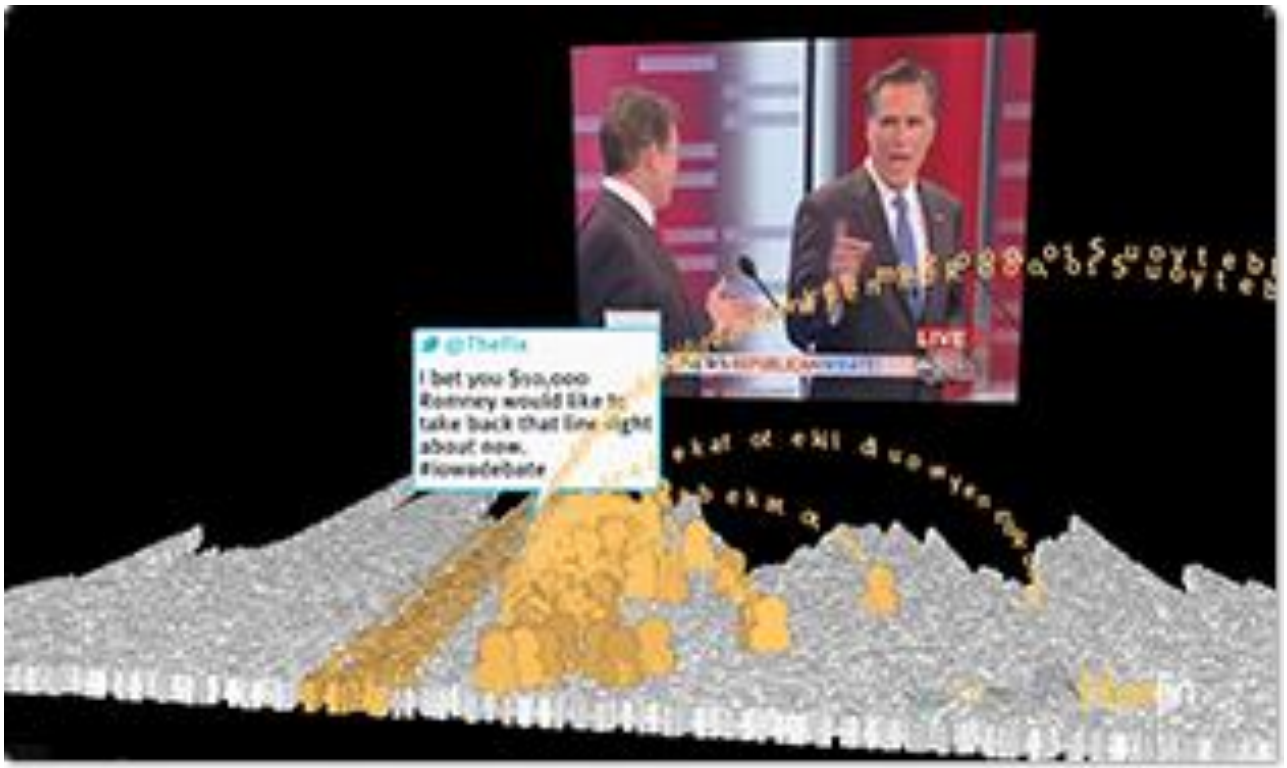
HARVARD UNIVERSITY §

http://www.ccs.neu.edu/home/amislove/twittermood

Social Media and Crises

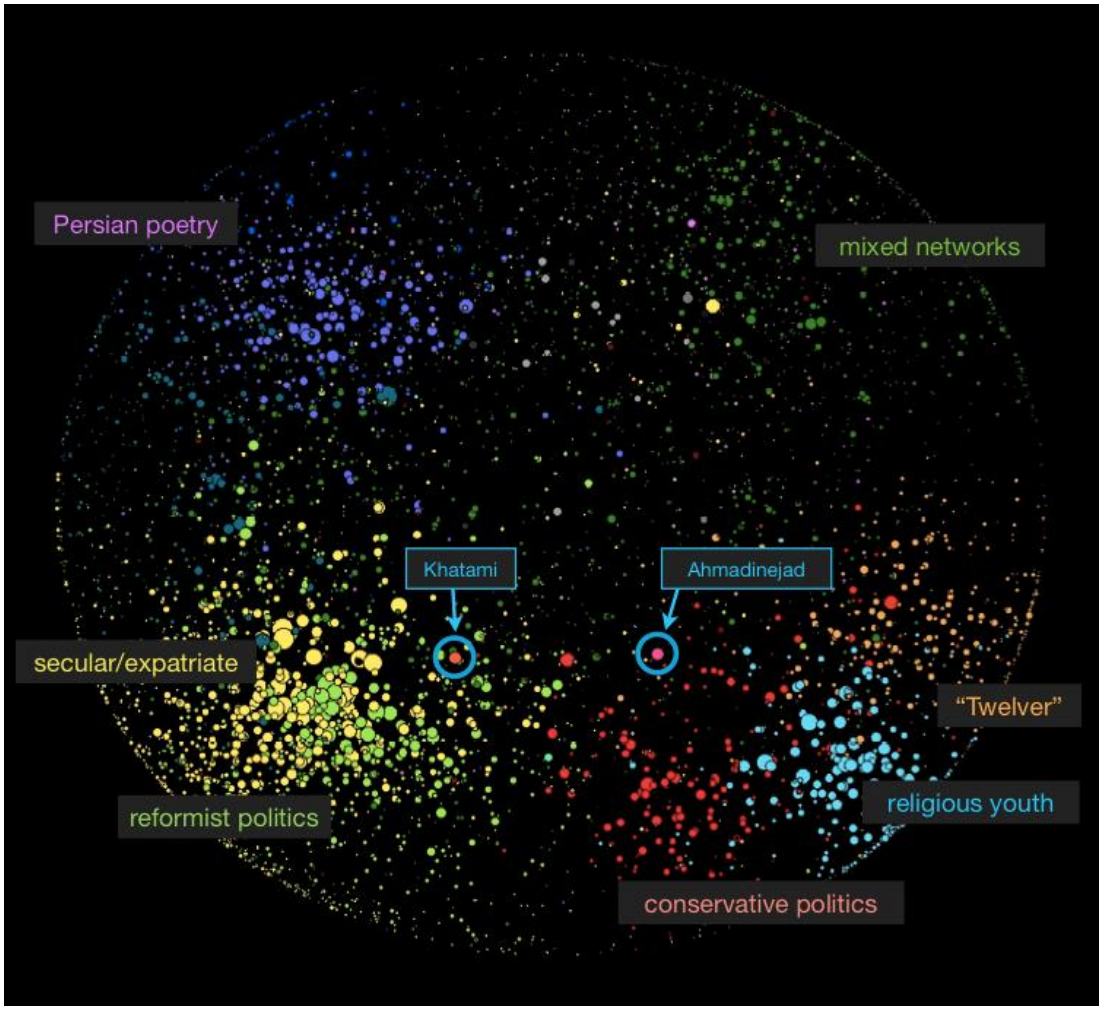


Mislove et al. 2011



Bluefin Technologies

Computational Social Science



John Kelly, Berkman center Harvard

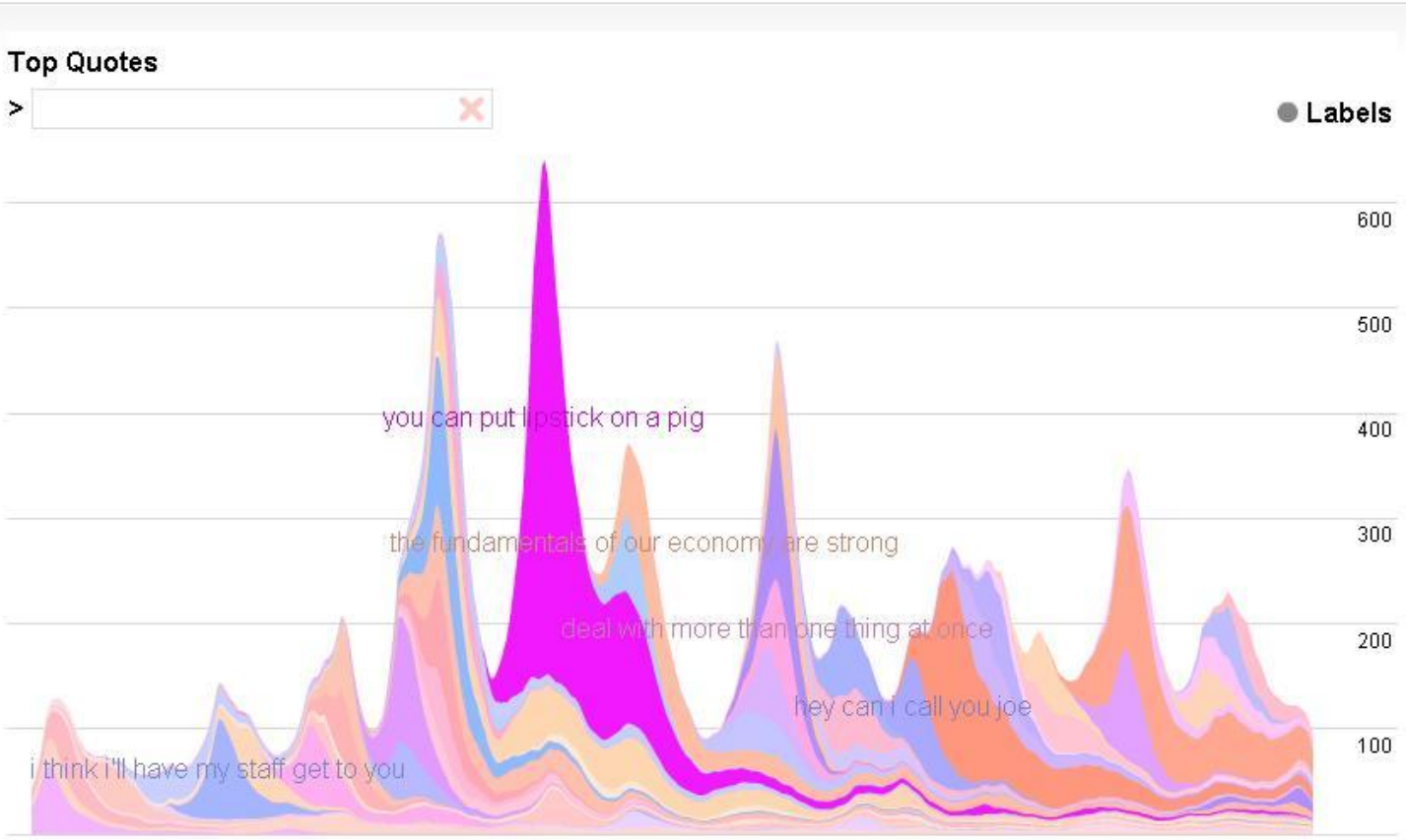
MemeTracker

by Jure Leskovec, Lars Backstrom and Jon Kleinberg

Navigation:

- Home
- About
- Time lag of media on reporting a story
- Gallery

Most mentioned phrases over last three months



Leskovec, Backstrom, Kleinberg

This course will…

cover a core of **algorithms** for mining behavioral data.

explore and develop **tools** for behavioral data mining, addressing **ease-of-use** and **scalability**.

describe **models of human behavior** that shape algorithm design and implementation.

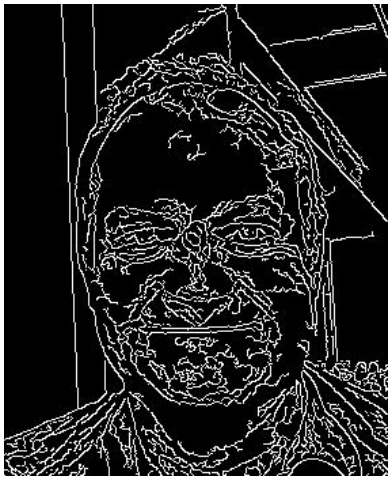
have several short coding/modeling assignments and one major project (individual or team of two).

Behavior Change

Proceeds by charting **attitudes** toward change topics, discovering the apparent causal influence of those attitudes, and trying to change the ones that are obstructing change.

At a deeper level peoples' ability to change depends on an interconnected set of schema about themselves. These have been explored through **identity narratives.** Journal and diary sites provide and ideal resource to analyze those patterns.

Instructor



John Canny (Ph.D. MIT 1987)
research in computer vision, robotics

DataMining projects

- MarkLogic co-founder 2003
- BigTribe/Overstock 2004-2005
- Yahoo 2007-2008
- Ebay 2009-2010
- Quantcast 2010-2011

Research in behavior change, health care, education, computational social science: using data mining techniques on social media.

Course Mechanics

The class wiki is up at:
http://bid.berkeley.edu/cs294-1-spring12/index.php/Main_Page

There is no textbook for the course, but several recommended texts are up on the wiki.

There will be required and recommended readings posted before each class. Those are the "text" for the class.

There is a Piazza for Berkeley as "CS294-1". Please sign up on
http://piazza.com

TA, Office Hours, Section

Teaching Assistants

- Kenghao Chang: (BID Lab, 354 Hearst Mining)

Office Hours

- John Canny: Tu 4-5, in 637 Soda Hall, and by appointment

Section

- Weds 1-2pm in 310 Soda

Prerequisites

A desire to learn a bunch of new topics and the time to do it.

In order:

- Basic machine learning:
  - Linear algebra, Bayesian statistics
- Languages: Scala, Xquery, Hadoop
  - Matlab/R familiarity will help
- Characteristics of human behavior in the large
  - Power laws, personality, diffusion theory, dual-process models?
- Some aspects of system/cluster architecture that most affect performance.

Course Size

The course size is limited by the room and by cluster resources, at least this time around.

The good news is that CS281B (Alex Smola) has a very similar emphasis this semester. i.e. machine learning at scale.

What is data mining?

Data Mining

The process of exploiting large amounts of data? – YES

The art of discovering structure and value in "messy" data? – YES!, and this is where innovation happens.

- Agile **visualization and analysis** of raw data
- Quick **hypothesis testing** with "sketch" models
- Ability to **test against samples** of big datasets
- Ability to easily **migrate to large scale**
- Ability to **get performance data** on the scaled-up model

Tools that are out there

- **Matlab** and **R**: basic math and matrix algebra + toolboxes + user-contributed m-files
- **Hadoop** Map-Reduce + Java
- ML toolkits: **Weka**, Apache **Mahout** (for Hadoop)
- **MarkLogic Server** (Xquery + XML datastore) + open-source counterparts: eXist, BaseX,…
- **Microsoft SQL server** with Analysis Services

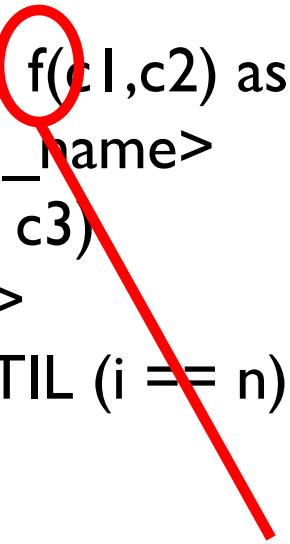
Tools Wish List

- Simple, SQL-like interface (like Hive, which sits on Hadoop) + matrix algebra
- Good numerical performance (like Matlab) but a real programming language (SciPy)
- Add some new operations to MR to support machine-learning algorithms (reduction trees, shared memory, early termination…)

Tools Wish List

Something like

SELECT c1, c2, f(c1,c2) as c3 FROM <table1>
WITH <model_name>
UPDATE g(c1, c3)
INTO <table3>
ITERATE_UNTIL (i == n)
SKETCH 0.1
ORDERBY…
GROUPBY…



Mapper and reducer functions with same name

On in-memory or Hadoop tables (c.f. Spark) or ML data

- Runs on a Java Virtual Machine, **full access** to Java code, and mostly vice-versa (can run in Hadoop).

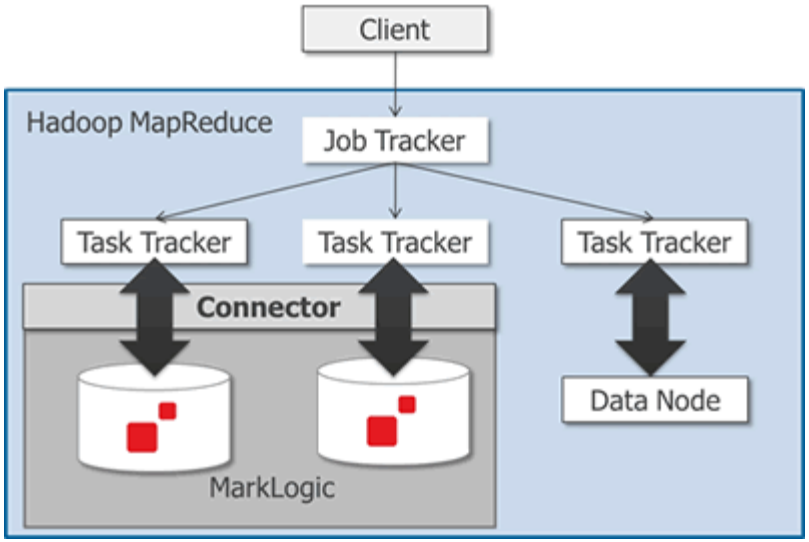
- Has a **REPL** (Read-Eval-Print Loop) so with the right libraries it's a Matlab-like analysis/viz environment.

- Lots of language virtues:
  - Modular syntax, can add operators
  - Smart type inference, implicit type conversions
  - Functions are first-class objects, has anonymous functions
  - "Mapping" operations widely used in programs
  - In fact, can realize map-reduce operations on tables in a very natural way.

Tools we will use

ScalaNLP, a Berkeley-developed toolkit for Natural-language processing and machine learning, and a simple toolkit for matrix algebra **(BIDMat)**.

We will work with **Hadoop** for large datasets, and **MarkLogic Server** for live querying. MarkLogic has a Hadoop connector that bridges between the two clusters.



Developing New Tools

Intentional side-effect of the course.

- We will be developing libraries* for local and distributed data mining as we go along, layering on Hadoop, Scala and MarkLogic.

* which in the Scala context means interactive environments

A Few Words on Performance

- **Accuracy:** as scientists, we care most about accuracy: drawing the most accurate conclusions we can from the data.

- **Resolution:** we also want to be able to tell whether putative structure is there at all, i.e. can we resolve it?

- For exploring data we **want answers quickly**!

Generally speaking, both accuracy and resolution improve with the amount of data processed. So there are many reasons to improve speed and scale, i.e**. algorithm and system performance** really matter.

Simple Performance Rules

Performance is a small but very important part of the course. You don't need to be a systems specialist to be able to apply these principles.

- Understanding asymptotics, $O(n^2)$ vs $O(n \log n)$
- Understanding memory: sequential vs. random access, cache hierarchy.
- Having a feel for constants: hashing, string comparison, array access etc.

Often, the solution is to use an efficient library (e.g. Intel MKL or Atlas for matrix operations, scientific functions, and random numbers).

Case Study: Grandma's Matrix Multiply

In Java: multiply C = A * B,

where A is an m*p matrix, B is a p*n matrix

```
for (int i = 0; i < m; i++) {
  for (int j = 0; j < n; j++) {
    double sum = 0;
    for (int k = 0; k < p; k++)
      sum += A[i,k] * B[k,j];
    C[i,j] = sum;
}}
```

(For Java, we map [i, j] to [i + j*nrows])

Case Study: Matrix Multiply

Performance: Mahout (1000x1000 multiply, Intel i5):

1.2 secs, 0.16 Gflops

Matlab performance (i.e. Intel MKL), same machine, same problem: ??

Case Study: Matrix Multiply

Performance: Mahout (1000x1000 multiply, Intel i5):

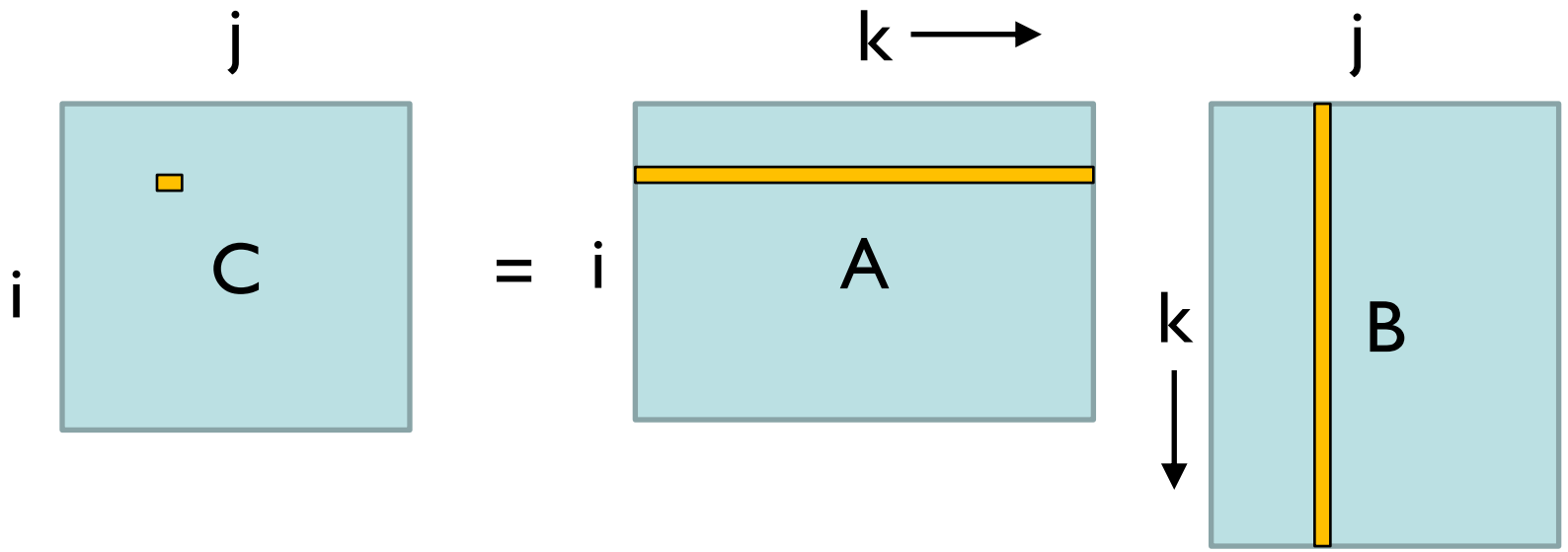
1.2 secs, 0.16 Gflops

Matlab performance (i.e. Intel MKL), same machine, same problem: 26 msecs, 75 Gflops, about **500x** faster.

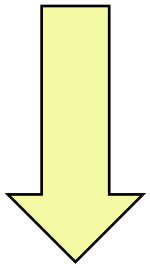
Where does the difference come from?:

- Matlab's implementation is multi-threaded, 4x
- 2x Overhead in array access calculations "getQuick(i,j)"
- * Memory "grain" (about 10x).
- Memory hierarchy (L1, L2, L3 caches), Intel SSSE4 instructions, loop unrolling, hand assembly-coding,…

Case Study: Matrix Multiply

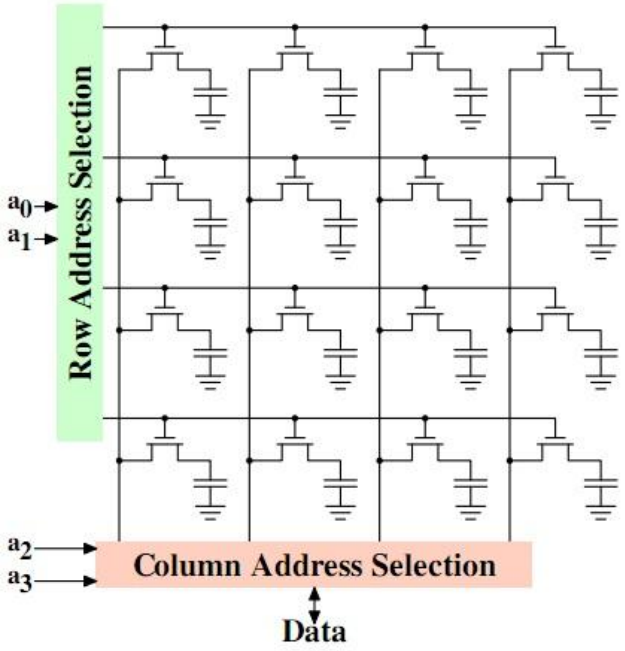


Column-major order, memory "grain"



DRAM

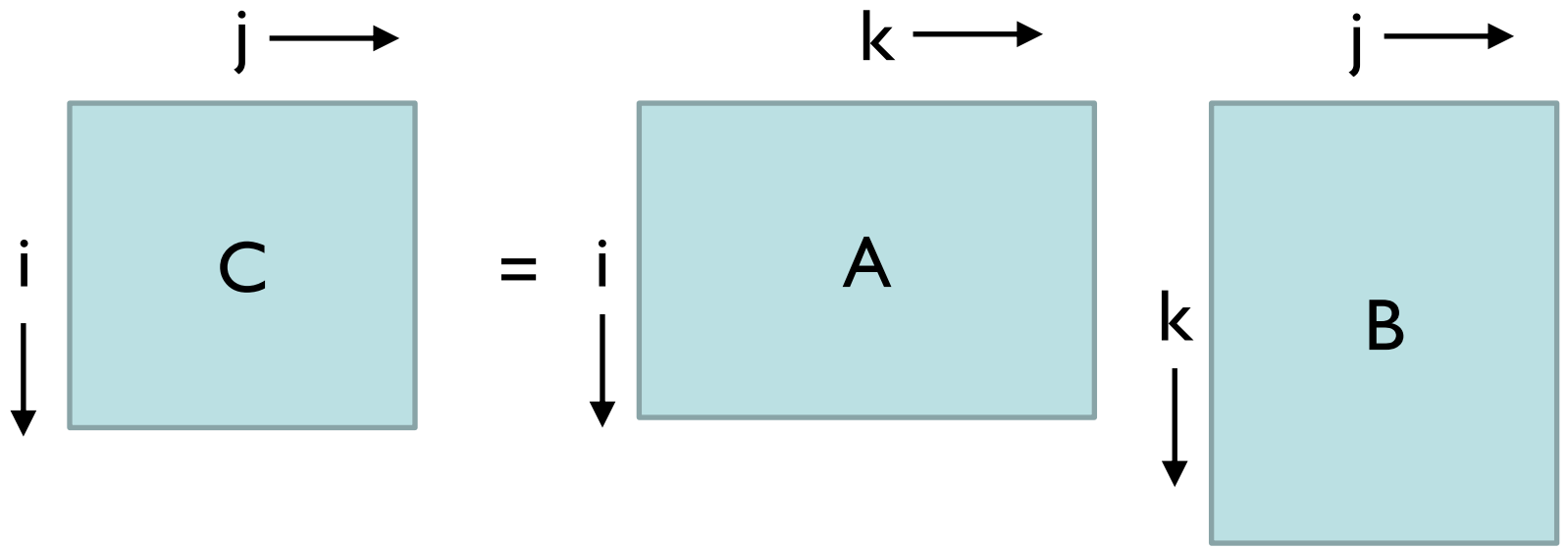
- Reading one location really reads a row (16kB for 4GB DRAM). Sequential access >>faster>> random access



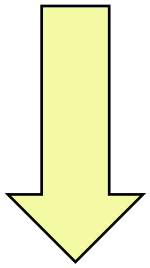
16 kB row

Figure 2.7: Dynamic RAM Schematic

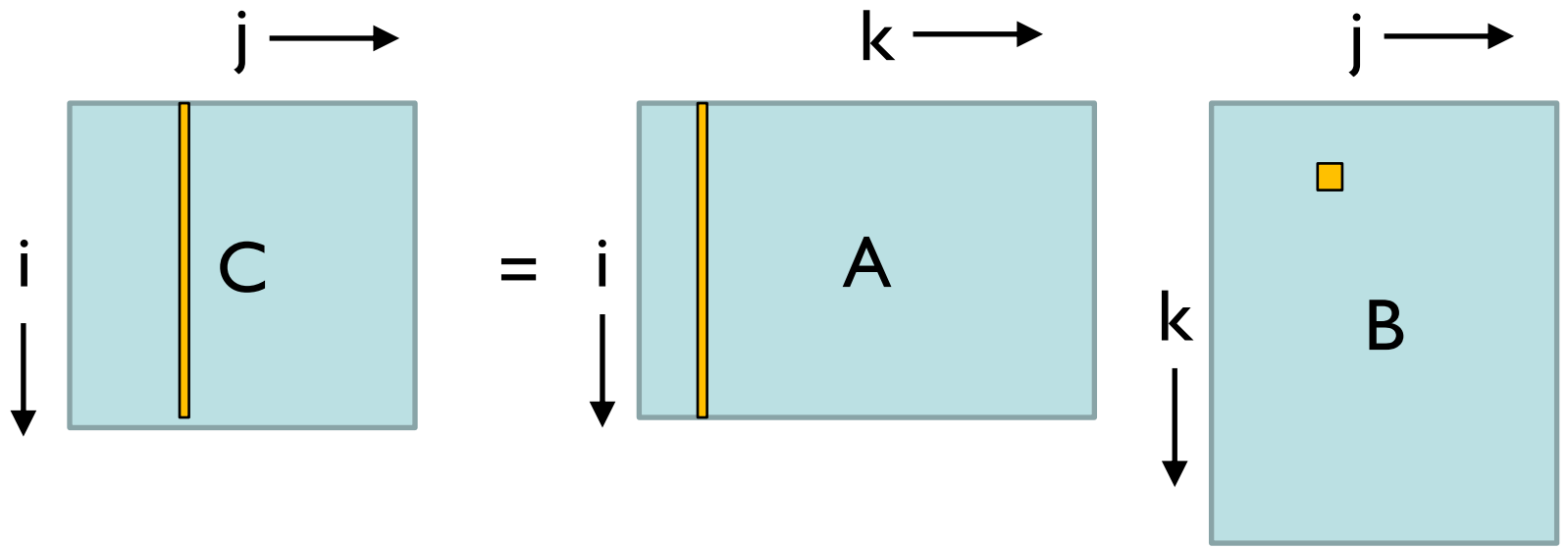
Fixing Matrix Multiply



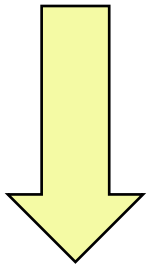
Column-major order, memory "grain"



Fixing Matrix Multiply



Column-major order, memory "grain"



Case Study: Matrix Multiply

(Assume C is initialized to zero)

```
for (int j = 0; j < n; j++)
  for (int k = 0; k < p; k++) {
    double bval = B[k,j];
    for (int i = 0; i < m; i++)
      C[i,j] += A[i,k] * bval;
}
```

About **10x** faster on 1000x1000 matrices, i7 processor. Caveats:

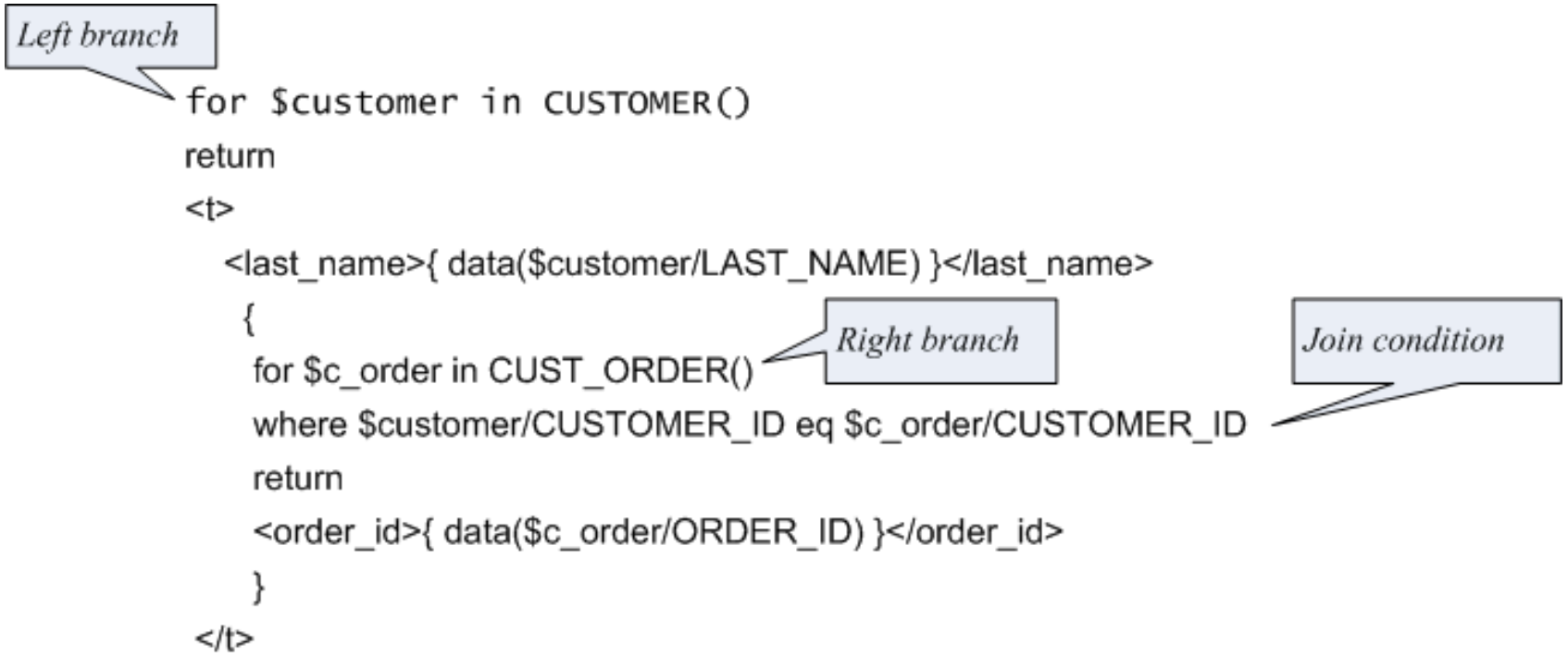
- 100x100 multiply also much faster with Grandma's alg.!
- But you can get similar gains with large dense/sparse operations by paying attention to grain.

Aside: Dense-Sparse Matrix Multiply

Is the limiting step in many algorithms (LDA, regression,…). There are probably some gains to be had by exploiting the structure (Power-law distribution of term frequencies) of the sparse matrices.

Real-Time vs. Offline Mining

Or: what is XQuery/MarkLogic and what is it good for?



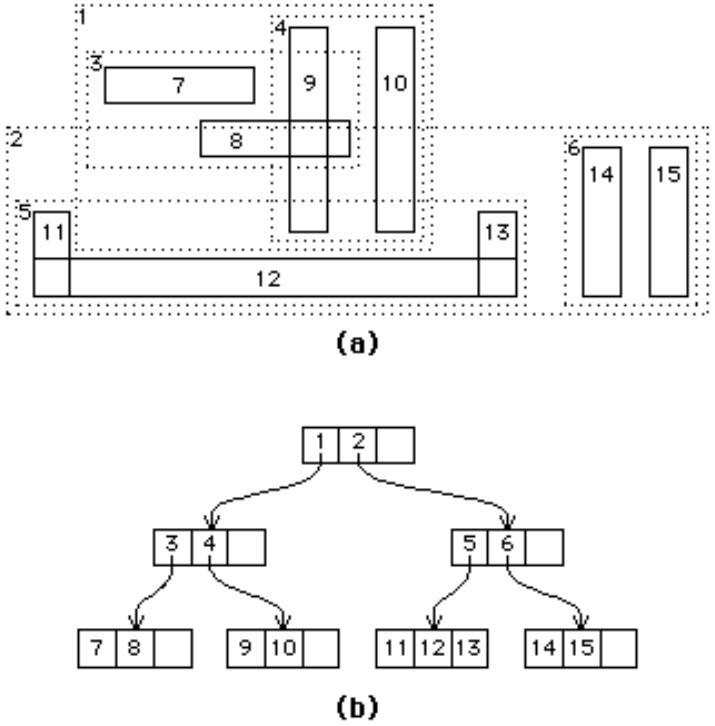
XML vs. relational DBs

- XML stored natively in tree structure.
- Schema are trees, can be recursive.
- Indices on schema elements.
- Can construct new kinds of indices from builtins.
- Queries works directly with HTML, web services, SOAP etc.

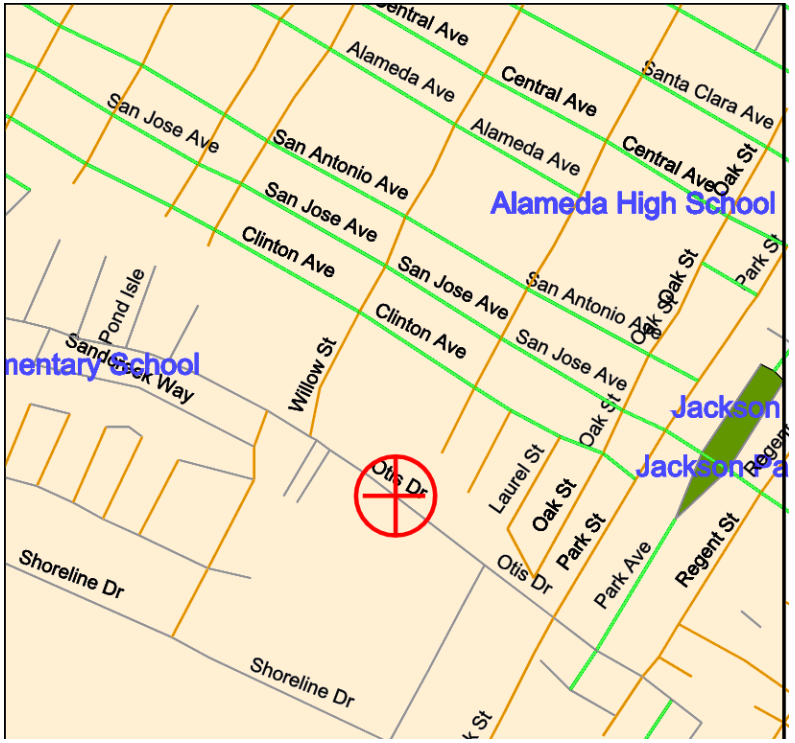
- Data types stored as columns in a table.
- Schema define columns and relations.
- Indices on columns.

- Indices only for known datatypes.
- “code” written separately in PHP, java etc.

An example



Flat geo-data (from Census) stored hierarchically as R-trees in XML



An SVG (XML) map generated with an XQuery on the geodata

Sentiment Analysis on ML

Raw tweets, labeled from overall word content:

```
<status … sentiment="0.3">
<status … sentiment="-0.45">
<status … sentiment="0.2">
```

Search for "Romney" ➔

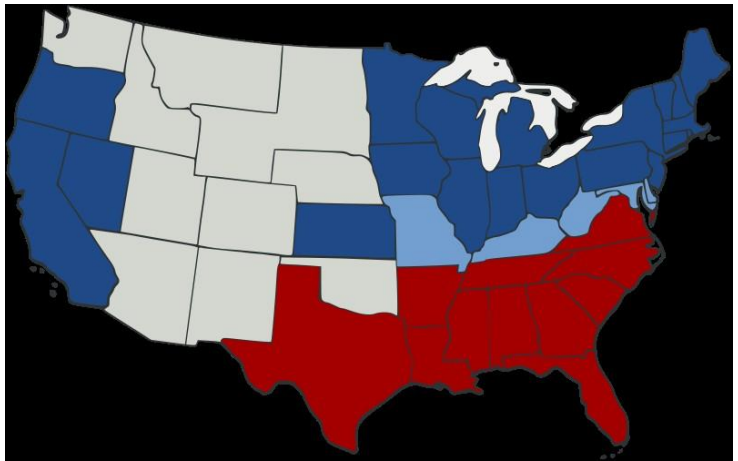
```
<status text="…Romney…" sentiment="0.2">
<status text="…Romney…" sentiment="0.3">
```

Average by state ➔

```
<sentiment state="CA" score="0.1">
<sentiment state="WA" score="0.2">
```

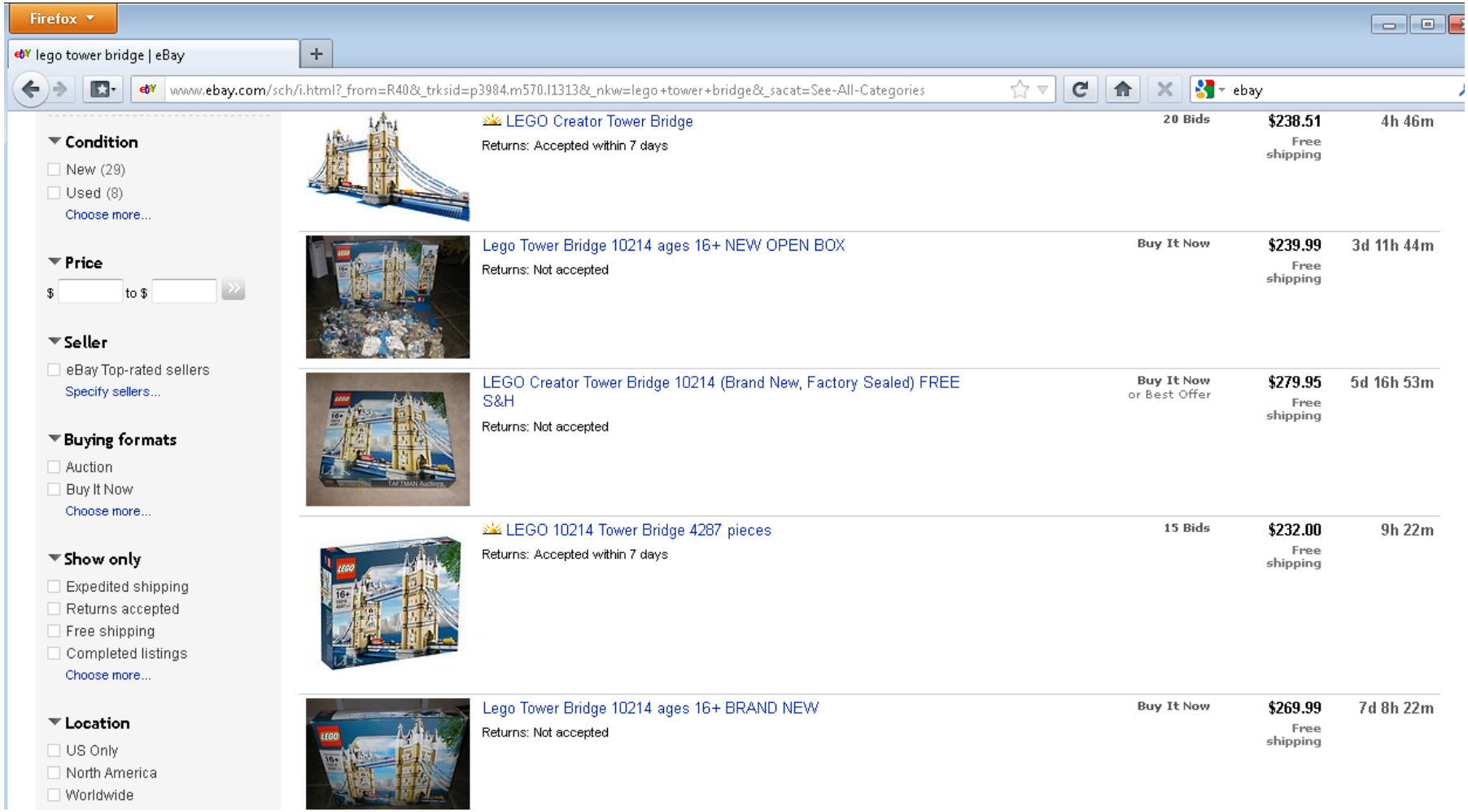
Join with SVG map :

➔



Real-Time Clustering (SIGIR 2011)

Ebay recommendations: Need to identify products first



Real-Time Clustering

Ebay recommendations:

- Identify products by clustering
- Build a hierarchy of naïve Bayes models for products-to-products
- When new items come in, they are assigned to a product cluster so they can source or sink recommendations.

The most “sellable” items are short-lived. The quicker they are sorted into product categories, the better the odds of their being sold quickly. That improves recommender revenues.

Generative Clustering

- An item $\mathbf{x}$ is a 3-tuple of vectors: $\mathbf{x} = (\mathbf{b}, \mathbf{c}, \mathbf{g})$.
  1. For binary variables: $b_v \sim \text{Binom}(p_v), \forall v \in \{1, \ldots, V\}$;
  2. For categorical variables: $c_u \sim \text{Mult}(\theta_u), \forall u \in \{1, \ldots, U\}$;
  3. For continuous variables: $g_s \sim \mathcal{N}(\mu_s, \sigma^2), \forall s \in \{1, \ldots, S\}$.
- Given a latent product $\mathbf{z}_k$, the likelihood of an item $\mathbf{x}_i$ is:

$$p(\mathbf{x}_i|\mathbf{z}_k) = \prod_{v:b_{iv}=1} p_{kv} \prod_{v:b_{iv}=0} (1 - p_{kv}) \prod_u \theta_{ku}$$

$$\times \prod_s \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(g_{is} - \mu_{ks})^2}{2\sigma^2}\right).$$

Real-Time Clustering

Probabilities of membership in each cluster are non-linear, but log likelihoods simplify to:

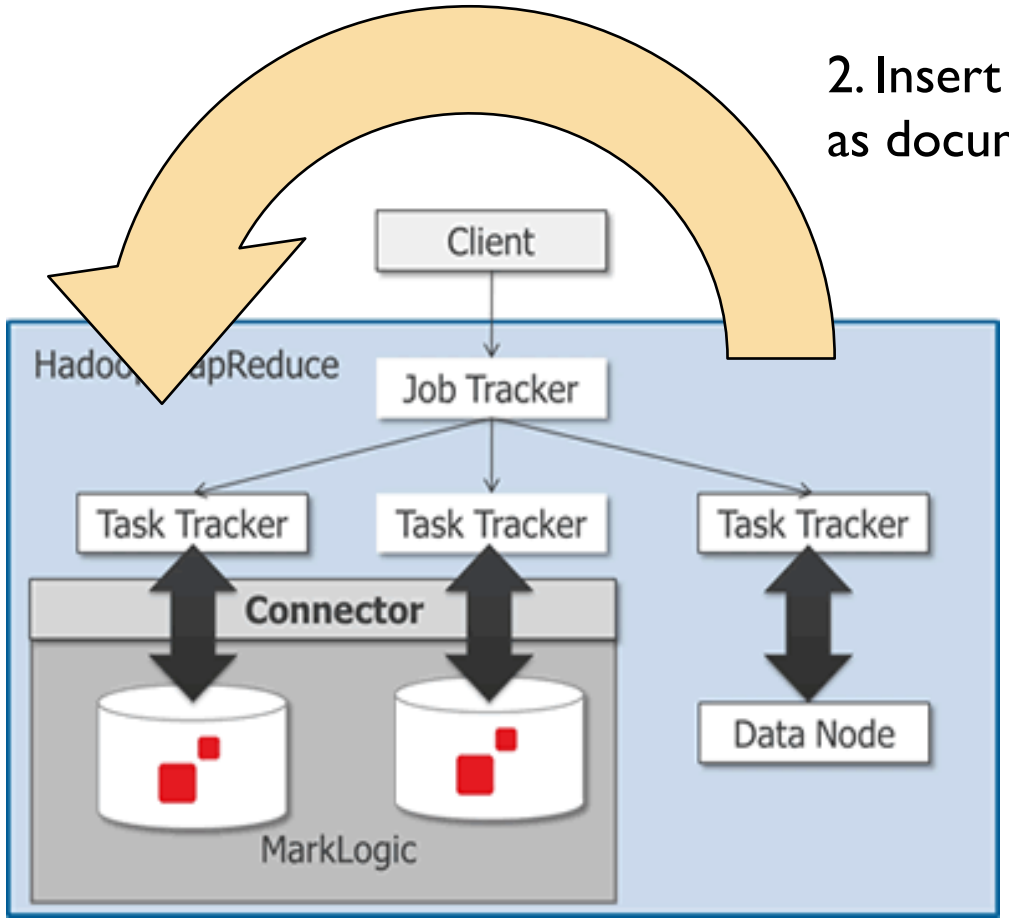
$$\ell(\mathbf{x}_i|\mathbf{z}_k) \propto \sum_v a_{kv} b_{iv} + e_k,$$

where a_{kv} are transformed cluster centers, b_{iv} are item descriptions.

i.e. finding the best-matching cluster reduces to finding the largest inner product between item description vectors and cluster centers. Treat cluster centers as "documents," and this becomes a search problem.

Real-Time Clustering

2. Insert cluster centers as documents



3. New item description text query

1. Compute 10-100k clusters

4. Best matching clusters returned in order