

Data mining

Rasmus Pagh

Reading: RG 26

Today

- Brief introduction to data mining.
- Association mining:
 - What are associations?
 - A priori algorithm
 - Sketching algorithms

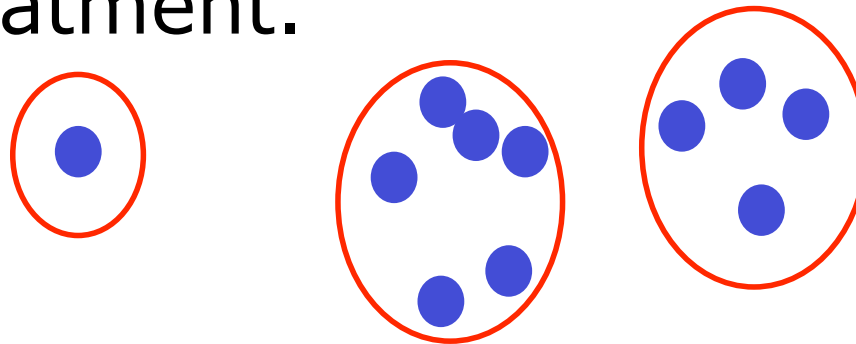
Data mining: Many flavors

- Statistics.
- OLAP/dimensional modeling: Complex aggregation queries over possibly huge data sets ("decision support").
- Machine learning and artificial intelligence.

- Here: "Discovery of useful summaries of data" and "Discovery of useful observations about data".

Examples of data mining queries

- **Clustering.** Group objects together in clusters of "similar" objects, e.g., customer groups that need different treatment.



- **Association rules.** Find "interesting" correlations in data. Amazon.com: "Customers who bought this book also bought ..."

Examples of machine learning

- **Inferring decision trees.** Find a decision tree that classifies most objects correctly (supervised learning).
- **Learning to classify images.** Newest approaches combine classical machine learning with clustering techniques. (Talk by Zhi-Hua Zhou this Tuesday.)
- **Learning to diagnose from images.** Combination of large image databases, image analysis, and machine learning techniques. (Mads Nielsen, KU.)

An exploding area

Corporate acquisitions of Business Intelligence (BI) vendors alone last year totaled over 10 billion dollars, and that is only for the “front end” of the data analysis toolchain.

Claremont Report on Database Research, SIGMOD Record, September 2008.

Today: Market baskets

- Data is a sequence “baskets”, where each basket is a set of items.
- Want to derive association rules such as “80% of the customers who buy corn flakes also buy milk”. (confidence is 80%)
- Note that this rule is only **interesting** if the percentage of customers buying milk is significantly different from 80%.
- Observation: An interesting rule requires the pair of item to appear frequently together.

Example

From slides by
Jeff Ullman

- Items = {milk, coke, pepsi, beer, juice}.
- Support threshold: We want to find item sets that occur in at least 3 baskets.

B1 = {m, c, b}

B2 = {m, p, j} **Relational**

B3 = {m, b}

B4 = {c, j} **representation?**

B5 = {m, p, b}

B6 = {m, c, b, j}

B7 = {c, b, j}

B8 = {b, c}

- Frequent itemsets: {m}, {c}, {b}, {j}, {m, b}, {c, b}, {j, c}.

Finding frequent items

- Simply count occurrences of all items.
- Easy if internal memory is large enough to hold all **distinct** items. (Why?)
- **Short problem session:**
In the general case, where there is not sufficient internal memory, how does one count number of occurrences?
(**Hint:** How is GROUP BY processed?)

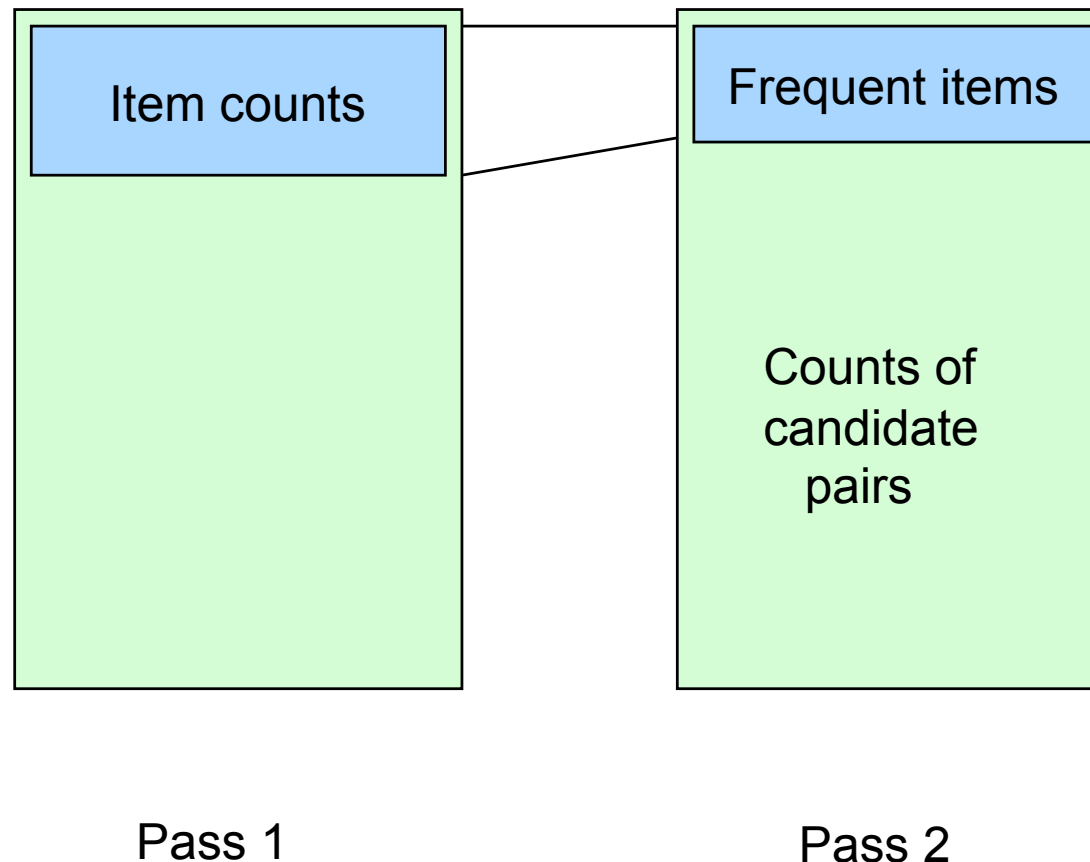
A Priori Property

- Observation (Agrawal & Srikant '94):

If an item set occurs c times, all its subsets occurs at least c times.
- In particular: To find frequent item sets of size 2, we only need to consider pairs of frequent items.

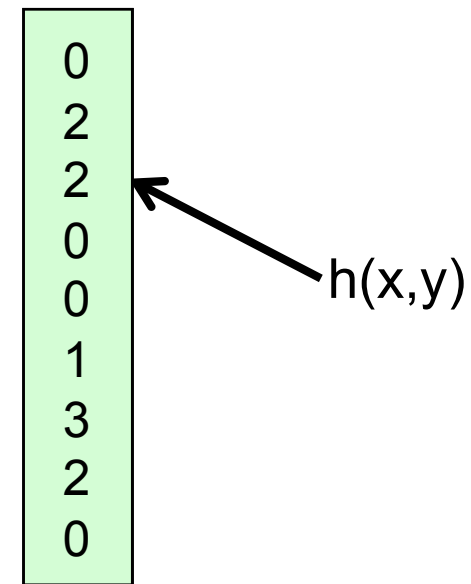
Picture of A-Priori Algorithm

From slides by
Jeff Ullman



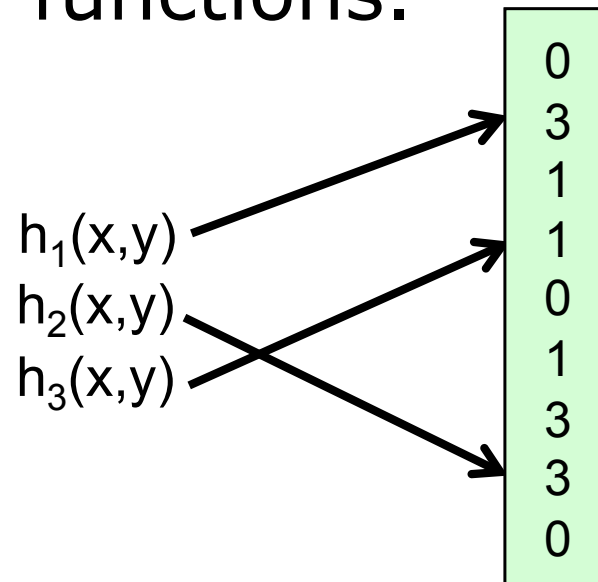
Counting things in less space

- Even if we do not have space for all pairs, we can still do *approximate* counting ("counting Bloom filter").
- **Idea:** When (x,y) is seen, increase the counter at $A[h(x,y)]$, where h maps to indexes of an array A .
- Key property: $A[h(x,y)]$ is an *upper bound* on the number of occurrences of (x,y) — can be used to deduce that some pair is *not* frequent.



Counting a little more precisely

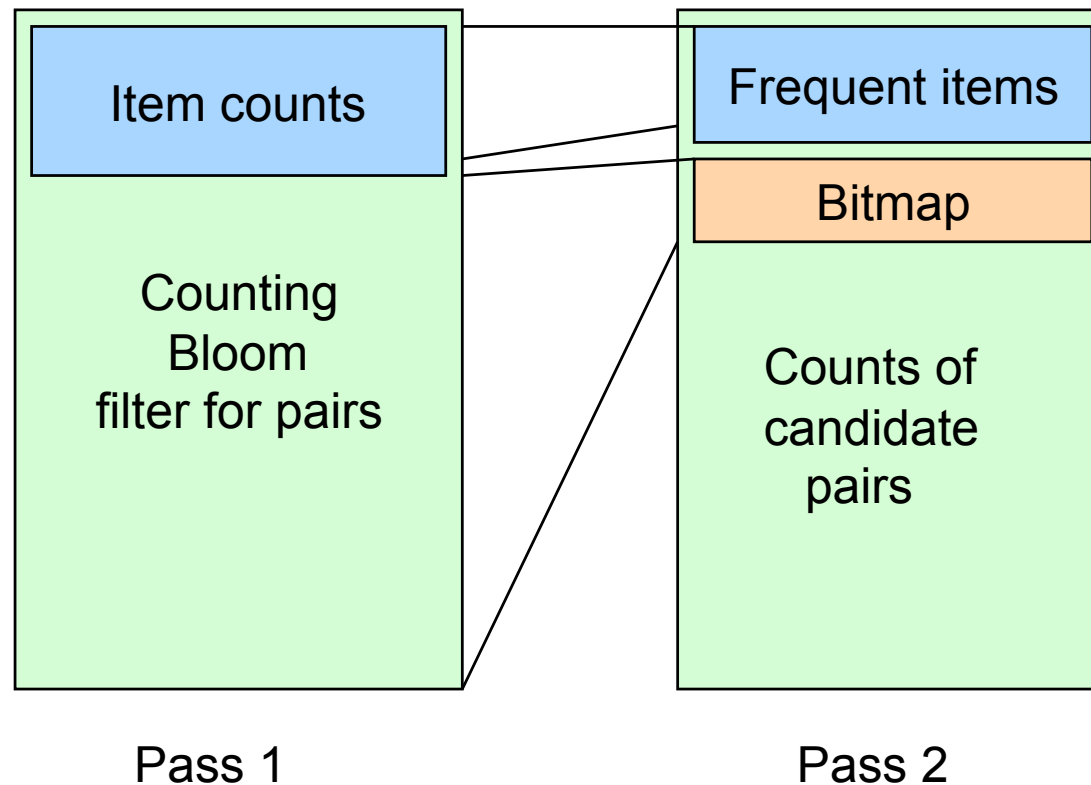
- Get more precise numbers if we use several hash functions.



- Key property: **Each** number $A[h_i(x,y)]$ is an upper bound on the number of occurrences of (x,y) .

Using a counting Bloom filter

Adapted from
slides by
Jeff Ullman



Need to count fewer pairs in pass 2

Saving more: Sampling baskets

- If the support threshold c is high, space and time usage can be reduced by sampling, say, a fraction $100/c$ of baskets.
- Then we expect to see pairs with at least c occurrences 100 times or more.
- Do exact counting only for pairs seen, say, 80 times or more.
- Drawback: We may miss some frequent itemsets (false negatives).

Summary

- The A-Priori algorithm finds frequent pairs, which are candidates for interesting relationships.
- The book discusses several extensions of basic association mining.
- Space usage of A-Priori can be high – we saw some ways of relieving that.
- **Next:** Exercise that, among other things, considers how A-Priori can be implemented with any amount of internal memory.

Next week

- Some examples of database research done at ITU.
- For example, we have recently found a completely new and very efficient way of doing association mining.
- Special guest star:
PhD student Rasmus Resen Amossen, just back from his stay at the database group at Yale.