Mining Data Streams III
Social Networks I

Alexander Schönhuth

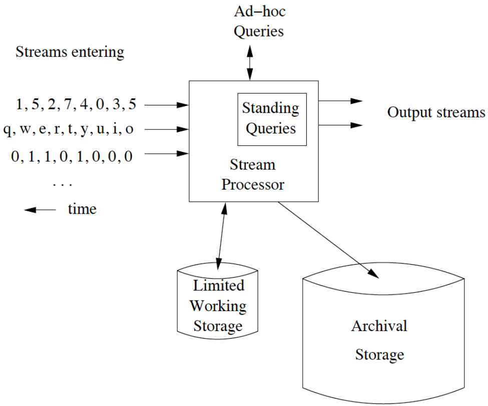
UNIVERSITÄT BIELEFELD
Faculty of Technology

Bielefeld University
June 22, 2023

Learning Goals Today / Overview

- *Mining Data Streams III*
  - Estimating Moments: Alon-Matias-Szegedy algorithm
- *Social Networks*
  - Intro: Social Networks are Graphs
  - How to Cluster Social Networks into Groups
  - Non-overlapping communities: the Girvan-Newman Algorithm

Data Stream Management System



A data stream management system

Adopted from `mmds.org`

Estimating Moments

The Alon-Matias-Szegedy Algorithm

Moments: Definition and Problem

Let $\mathcal{U}$ be the set of universal elements, that is each element of the stream is from $\mathcal{U}$. Assume that

- $\mathcal{U}$ is ordered, and
- its elements u_i are indexed by $1 \leq i \leq I$, where
- $I = |\mathcal{U}|$ is the cardinality of the universal set

k-th Moment

Consider a stream $x_1, ..., x_n$ where $x_j \in \mathcal{U}, j = 1, ..., n$

- Let $m_i := |\{j \in \{1, ..., n\} \mid x_j = u_i\}|$ be the count of u_i in the stream
- The *k-th order moment of the stream* is defined to be

$$\sum_{i=1}^{I} (m_i)^k \tag{1}$$

Moments: Examples

$$\text{k-th order moment:} \quad \sum_{i=1}^{I} (m_i)^k$$

Examples

- The 0-th moment of a stream is the number of *distinct* stream elements
- The 1-st moment of a stream is the *overall* number of stream elements
- The 2-nd moment of a stream is sometimes called the *surprise number*
  - Consider a stream of length 100, on 11 different elements
  - The most even distribution, 10 appearances for one particular element, and 9 for all others, yields surprise number $10^2 + 10 \cdot 9^2 = 910$
  - The most uneven ("surprising") distribution, 90 appearances for one particular element, and 1 for all others, yields surprise number $90^2 + 10 \cdot 1^2 = 8110$

Alon-Matias-Szegedy Algorithm: Notation

- Keeping a count for each element in main memory is infeasible
- Therefore, we need to *estimate* the k-th order moments
  ☞ The *Alon-Matias-Szegedy algorithm* does this

Notation:

- Let n be the length of the stream
- Let X be variables for which we store attributes
  - $X.element$ is an element of the universal set
  - $X.index$ is a position $1 \leq j \leq n$ where $X.element$ appears
  - $X.value$ is defined as the number of times $X.element$ appears in the stream between (and including) positions $X.index$ and n

Alon-Matias-Szegedy Algorithm: Notation

Example

Let the stream be $a, b, c, b, d, a, c, d, a, b, d, c, a, a, b$

- Stream length is $n = 15$
- The true second moment is $5^2 + 4^2 + 3^2 + 3^2 = 59$
- Let us keep three variables, X_1, X_2 and X_3, for which $X_1.index = 3, X_2.index = 8, X_3.index = 13$
- $X_1.element = c, X_2.element = d, X_3.element = a$
- $X_1.value = 3, X_2.value = 2, X_3.value = 2$

Along-Matias-Szegedy Algorithm: 2nd Moment

Alon-Matias-Szegedy Algorithm

- As estimate for the 2nd-order moment, compute, for any X,

$$n(2X.value - 1) \tag{2}$$

- ☞ As many estimates as there are stream elements!
- General strategy for using several X: average single estimates

Questions:

- Which one is the best?
- Should we better average several estimates at once?
- *What can we guarantee for any such estimate?*

Along-Matias-Szegedy Algorithm: 2nd Moment

Example (cont.): Stream = $a, b, c, b, d, a, c, d, a, b, d, c, a, a, b$

- We had $X_1.value = 3, X_2.value = 2, X_3.value = 2$
- $n(2X_1.value - 1) = 15(2 \cdot 3 - 1) = 75, n(2X_2.value - 1) = n(2X_3.value - 1) = 45$
- Yields average $(75 + 45 + 45)/3 = 55$, close to true value 59

Alon-Matias-Szegedy Algorithm: Theorem

The expected value $E(n(2X.value - 1))$ is defined as the average across *all* individual estimates $n(2X.value - 1)$.

Theorem

The 2nd-order moment

$$\sum_{i=1}^{I}(m_i)^2 \tag{3}$$

agrees with

$$E(n(2X.value - 1)).$$

Alon-Matias-Szegedy Algorithm: Proof I

- Let $e(j)$ be the stream element appearing at position j
- Let $c(j)$ be number of times $e(j)$ appears between (and including) positions j to n
- In example stream from above

$$a, b, c, b, d, \underbrace{\overset{e(6)}{a}, c, d, a, b, d, c, a, a, b}_{\text{4 appearances of } a \text{: } c(6)=4}$$

e.g. $e(6) = a$ and $c(6) = 4$

- For $X.index = j$
  - $e(j)$ corresponds to $X.element$
  - $c(j)$ corresponds to $X.value$

Alon-Matias-Szegedy Algorithm: Proof I

- $e(j)$ corresponds to $X.element$ for $X.index = j$
- $c(j)$ corresponds to $X.value$ for $X.index = j$

Inserting this in the definition of $E(n(2X.value - 1))$, one obtains

$$E(n(2X.value - 1)) = \frac{1}{n} \sum_{X} n(2X.value - 1) = \frac{1}{n} \sum_{j=1}^{n} n(2c(j) - 1) \tag{4}$$

by canceling factors further simplifying to

$$E(n(2X.value - 1)) = \sum_{j=1}^{n} (2c(j) - 1) \tag{5}$$

Alon-Matias-Szegedy Algorithm: Proof II

Regroup summands by their associated values $e(j)$:

$$\sum_{j=1}^{n}(2c(j)-1) = \sum_{a}\sum_{j:\ e(j)=a}(2c(j)-1) \tag{6}$$

Consider one particular a, let m_a be count of a in stream:

- Last j where a appears: $2c(j)-1 = 2\times 1 - 1 = 1$
- Second last j where a appears: $2c(j)-1 = 2\times 2 - 1 = 3$
- $\vdots$
- First j where a appears: $2\times m_a - 1$

Alon-Matias-Szegedy Algorithm: Proof III

Consider one particular a, let m_a be the number of times a appears in stream:

- Last j where a appears: $2c(j) - 1 = 2 \times 1 - 1 = 1$
- Second last j where a appears: $2c(j) - 1 = 2 \times 2 - 1 = 3$
- $\vdots$
- First j where a appears: $2c(j) - 1 = 2 \times m_a - 1$

This yields

$$\sum_{j:e(j)=a} (2c(j) - 1) = 1 + 3 + 5 + \ldots + (2m_a - 1) \overset{(*)}{=} (m_a)^2 \tag{7}$$

where $(*)$ reflects a well-known equality from basic calculus.

Alon-Matias-Szegedy Algorithm: Proof IV

This yields

$$\sum_{i:e(i)=a} (2c(j)-1) = 1+3+5+\ldots+(2m_a-1) = (m_a)^2$$

where the last equation follows from an easy induction.

Overall,

$$E(n(2X.value-1)) \stackrel{(5)}{=} \sum_{j=1}^{n}(2c(j)-1) \stackrel{(6)}{=} \sum_{a} \sum_{j:\ e(j)=a} (2c(j)-1) \stackrel{(7)}{=} \sum_{a}(m_a)^2 \quad (8)$$

which concludes the proof. □

Estimating Higher-Order Moments

- Observation: Adding $2v-1$ for $v = 1, ..., m_a$ amounts to $(m_a)^2$
- The elementary proof makes use of the equation: $2v - 1 = v^2 - (v-1)^2$ which can be exploited using the "telescope property":

$$\begin{aligned} & 2m_a - 1 + 2(m_a - 1) - 1 + ... \\ = \ & m_a^2 \underbrace{-(m_a-1)^2 + (m_a-1)^2}_{=0} \underbrace{-(m_a-2)^2 + (m_a-2)^2}_{=0} - ... \\ = \ & m_a^2 \end{aligned} \tag{9}$$

- Analogously, by $v^3 - (v-1)^3 = 3v^2 - 3v + 1$:

$$\sum_{v=1}^{m_a} 3v^2 - 3v + 1 = (m_a)^3 \tag{10}$$

Estimating Higher-Order Moments

- We had

$$\sum_{v=1}^{m_a} 3v^2 - 3v + 1 = (m_a)^3 \tag{11}$$

- So, for a variable X, we can use

$$n(3((X.value)^2 - 3X.value + 1)) \tag{12}$$

as an estimate for the third order moment

- For arbitrary k, for a variable X, take

$$n((X.value)^k - (X.value - 1)^k) \tag{13}$$

as estimate for k-th order moment

Moments for Infinite Streams

- *Situation:* Stream length n grows with time
- *Problem:* Need to select variables X, such that $X.index$ is uniformly distributed
- So, selecting variables X a priori tends to be biased
  ☞ non-uniform
- *Solution:* Maintain as many variables as possible. As stream grows:
  - Discard existing variables
  - Replace by new ones
  - such that at all times, variables are uniformly distributed

Moments for Infinite Streams

- *Solution:* As stream grows:
  - Replace existing variables by new ones
  - such that at all times, variables are uniformly distributed
- *Remark:* This establishes a generally applicable strategy for sampling elements from a stream:
  - Recall the problem of selecting representative samples
  - Recall the general sampling problem

Moments for Infinite Streams: Solution

- Suppose we can store/maintain s variables
- Suppose we have seen n stream elements
- Suppose the s different $X.index$ are uniformly distributed
- That is, the probability to see position $1 \leq j \leq n$ among the selected $X.index$ is s/n

Moments for Infinite Streams: Solution

- The probability to see position $1 \leq j \leq n$ among the selected $X.index$ is s/n
- Upon arrival of $(n+1)$-st element, do
  - Pick position $n+1$ with probability $s/(n+1)$
  - If picked, create variable X with $X.index = n+1$, and throw out any earlier X with equal probability $1/s$
  - If not picked, keep existing variables
- *Claim:* Afterwards, each position has been selected with probability $s/(n+1)$

Moments for Infinite Streams: Solution

- Upon arrival of $(n+1)$-st element, do
  - Pick position $n+1$ with probability $s/(n+1)$
  - If picked, create variable X with $X.index = n+1$, and throw out any earlier X with equal probability $1/s$
  - If not picked, keep existing variables
- *Claim:* Afterwards, each position has been selected with probability $s/(n+1)$

Proof:

- $(n+1)$-st position is picked with probability $s/(n+1)$
- Let $1 \leq j \leq n$ any other position: proof by induction
- Induction hypothesis: before $(n+1)$-st element arrived, j had been picked with probability s/n
- With probability $1 - s/(n+1)$, probability for having j stays s/n
- With probability $s/(n+1)$, probability for having j is $(s-1)/s$

Moments for Infinite Streams: Solution

Proof:

- $(n+1)$-st position is picked with probability $s/(n+1)$
- With probability $1 - s/(n+1)$, probability for having j stays s/n
- With probability $s/(n+1)$, probability for having j is $(s-1)/n$

Overall

$$(1 - \frac{s}{n+1})(\frac{s}{n}) + (\frac{s}{n+1})(\frac{s-1}{s})(\frac{s}{n}) \tag{14}$$

simplifying to

$$(1 - \frac{s}{n+1})(\frac{s}{n}) + (\frac{s-1}{n+1})(\frac{s}{n}) = ((1 - \frac{s}{n+1}) + (\frac{s-1}{n+1}))(\frac{s}{n}) \tag{15}$$

yielding

$$(\frac{n}{n+1})(\frac{s}{n}) = \frac{s}{n+1} \tag{16}$$

□

Social Networks as Graphs

Social Networks: Introduction

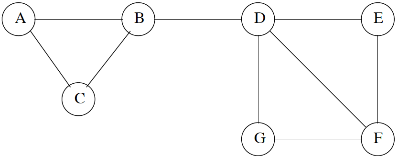
Basic Examples

- Facebook, Twitter, Google+

Defining Properties

- Collection of entities participating in network
  - Usually people, but other entities conceivable
- There is a relationship between the entities
  - Being friends is frequent relationship
  - Relationship can be of 0-1 type, or weighted
- Assumption of nonrandomness or locality
  - Hard to formalize, intuition is that relationships tend to cluster
  - If entity A is related with both B and C, B and C are related with larger probability

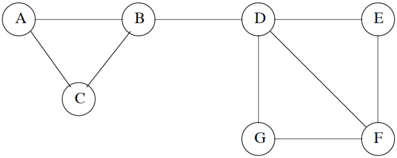
Social Network Graphs: Entities and Relationships



Adopted from `mmds.org`

- *Entities:* Nodes A to G
- *Relationships:* Represented by edges between nodes
  - *Example:* A is "friends" with B and C

Social Network Graphs: Locality



Adopted from `mmds.org`

- *Locality:*
  - There are 9 out of 21 possible edges: $\frac{9}{21} = 0.429$
  - Given nodes X, Y, Z such that there are edges $(X, Y), (Y, Z)$, random occurrence of (X, Z) is $\frac{7}{19} = 0.368$
  - However, across all pairs of existing edges $(X, Y), (Y, Z)$, probability that (X, Z) exists is $\frac{9}{16} = 0.563$

☞ Network exhibits locality

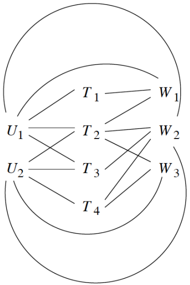
Social Networks: Examples

- *Telephone Networks:*
  - *Nodes* are phone numbers, *edges* exist if one number called another
  - *Edge weights:* Number of calls (within certain period of time)
  - *Communities:* Groups of friends, members of a club, people working at same company
- *Email Networks:*
  - *Nodes* are email addresses, *edges* indicate exchange of emails
  - *Edge directionality* may matter, so graph with directed edges
  - *Communities:* Similar to telephone networks

Social Networks: Examples

- *Collaboration Networks:*
  - *Nodes* e.g. represent authors, *edges* indicate working on same document
  - *Alternatively:* nodes represent documents, edges indicate that identical author contributed
  - *Communities:* Groups interested in / working on same subjects; documents sharing related content
- *Other:*
  - *Information networks:* Documents, web graphs, patents
  - *Infrastructure networks:* Roads, planes, water pipes, power grids
  - *Biological networks:* Genes, proteins, drugs
  - *Product co-purchasing networks:* E.g. Groupon

Several Types of Nodes



Adopted from `mmds.org`

Examples

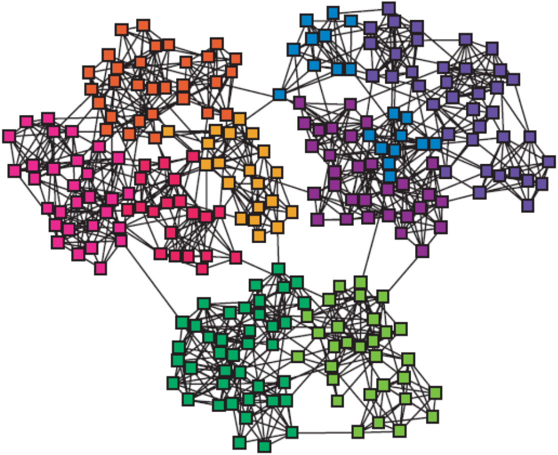
- Figure: Users (U) put tags (T) on web pages (W): tri-partite network
- Put documents and authors into one bi-partite network

Clustering Social Networks

Clustering Social Networks: Introduction

- An important aspect of social networks are *communities*
- Communities reveal themselves as groups of nodes that share unusually many edges
- Clustering social networks relates to the discovery of such communities

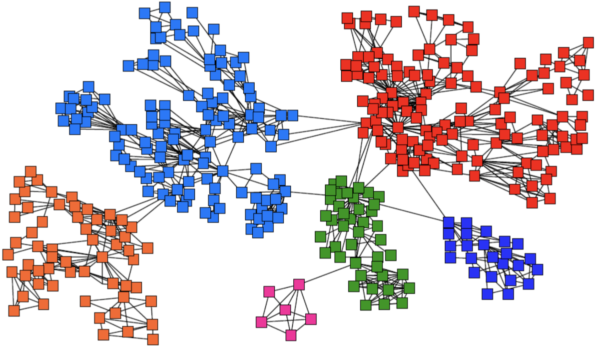
Communities



Differently Colored Communities in Social Network

Adopted from `mmds.org`

Clustered Network



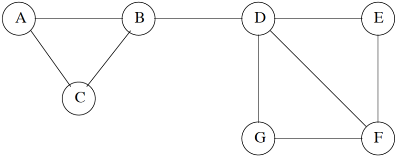
Differently Colored Clusters in Social Network

Adopted from `mmds.org`

Distance Measures in Social Networks

- Standard clustering techniques work with distance measures
- Distance measures are not obvious to define in social networks
  - Let $x, y \in V$ be two nodes in a social network $G = (V, E)$. The measure
    $$d(x,y) = \begin{cases} 0 & (x,y) \in E \\ 1 & (x,y) \notin E \end{cases}$$
    violates the triangle inequality, hence is no distance measure
  - Exchanging 0 with 1, and 1 with ∞ does not help
  - Other binary-valued measures (e.g. 1 and 1.5) agree with triangle inequality
- *But:* Additional issues apply

Social Networks: Clustering Issues



Communities: A-B-C and D-E-F-G

Adopted from `mmds.org`

- *Hierarchical Clustering:* Randomly picks closest nodes/clusters
- Distance between clusters: distance between closest points
- As soon as clusters are joined on B and D, clusters not as desired
- *Summary:* Standard clustering techniques difficult/impossible to sensibly implement

Betweenness

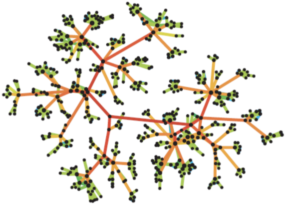
Idea: Identify edges that are least likely to be within community

Definition [Betweenness]

The *betweenness* of an edge (a, b) is

- the number of pairs of nodes (x, y) such that (a, b) makes part of the *shortest path* leading from x to y
- If for (x, y) there are several shortest paths, (a, b) is credited the fraction of shortest paths leading through (a, b) when computing its betweenness

Betweenness



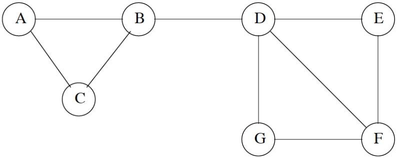
Telephone network:
Links between communities have great betweenness

Adopted from `mmds.org`

Explanation

- High betweenness means that (a, b) is a bottleneck for shortest paths
- If nodes (a, b) lie within community, there are too many options for shortest paths to circumvent (a, b) (so (a, b) gets credited only small fractions)

Betweenness: Example



Adopted from `mmds.org`

- (B, D) has the greatest betweenness, 12
  - It is on any shortest path between A, B, C and D, E, F, G
- (D, F) has betweenness 4
  - It lies on all shortest paths between A, B, C, D and F

General / Further Reading

Literature

- Mining Massive Datasets, 10.1, 10.2
  `http://infolab.stanford.edu/~ullman/mmds/ch10.pdf`
- Next lecture: "Social Networks II"; 10.3, 10.5 in *Mining of Massive Datasets*