Data Mining MTAT.03.183
Streams, Graphs, Web, DNA, SNA
Wrap-up

Jaak Vilo
2012 Fall
Machine learning is important
The “Dumb User” Perspective

Weka, RapidMiner, MSSSAS, Clementine, SPSS, R, …
Knowledge Discovery (KDD) Process

- This is a view from typical database systems and data warehousing communities
- Data mining plays an essential role in the knowledge discovery process
Summary so far

• Different types of data
• Data preparation (ETL, cleaning, ...)
• Statistics/significance
• Visualisation
• Large data – algorithmics, fast counting
• Queries/reporting, OLAP
• Machine learning
• Business value
Today

• Stream data
• Ordered sequences
• Pattern Discovery in sequences

• (Web) Information Retrieval
• Social Network Analysis
Streams, time series

- Time
- Sequence order and position
- Continuously arriving data

- Use or lose ...
• Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., Mining Data Streams: A Review, in ACM SIGMOD Record, Vol. 34, No. 1, June 2005, ISSN: 0163-5808


• Gama J., Knowledge Discovery from Data Streams, a book published by CRC Press, 2010.
Data Stream Mining is the process of extracting knowledge structures from continuous, rapid data records. A data stream is an ordered sequence of instances that in many applications of data stream mining can be read only once or a small number of times using limited computing and storage capabilities. Examples of data streams include computer network traffic, phone conversations, ATM transactions, web searches, and sensor data. Data stream mining can be considered a subfield of data mining, machine learning, and knowledge discovery.
• In many data stream mining applications, the goal is to predict the class or value of new instances in the data stream given some knowledge about the class membership or values of previous instances in the data stream. Machine learning techniques can be used to learn this prediction task from labeled examples in an automated fashion.

• In many applications, the distribution underlying the instances or the rules underlying their labeling may change over time, i.e. the goal of the prediction, the class to be predicted or the target value to be predicted, may change over time. This problem is referred to as concept drift.
Software

• **RapidMiner**: free open-source software for knowledge discovery, data mining, and machine learning also featuring data stream mining, learning time-varying concepts, and tracking drifting concept (if used in combination with its data stream mining plugin (formerly: concept drift plugin))
MOA (Massive Online Analysis): free open-source software specific for mining data streams with concept drift. It contains a prequential evaluation method, the EDDM concept drift methods, a reader of ARFF real datasets, and artificial stream generators as SEA concepts, STAGGER, rotating hyperplane, random tree, and random radius based functions. MOA supports bi-directional interaction with Weka (machine learning).
Literature on Stream Mining

Characteristics of Data Streams

- **Data Streams**
  - Data streams—continuous, ordered, changing, fast, huge amount
  - Traditional DBMS—data stored in finite, persistent data sets

- **Characteristics**
  - Huge volumes of continuous data, possibly infinite
  - Fast changing and requires fast, real-time response
  - Data stream captures nicely our data processing needs of today
  - Random access is expensive—single scan algorithm (*can only have one look*)
  - Store only the summary of the data seen thus far
  - Most stream data are at pretty low-level or multi-dimensional in nature, needs multi-level and multi-dimensional processing
Stream Data Applications

- Telecommunication calling records
- Business: credit card transaction flows
- Network monitoring and traffic engineering
- Financial market: stock exchange
- Engineering & industrial processes: power supply & manufacturing
- Sensor, monitoring & surveillance: video streams, RFIDs
- Security monitoring (intrusion detection)
- Web logs and Web page click streams
- Massive data sets (even saved but random access is too expensive)
DBMS versus DSMS

- Persistent relations
- One-time queries
- Random access
- “Unbounded” disk store
- Only current state matters
- No real-time services
- Data at any granularity
- Assume precise data
- Access plan determined by query processor, physical DB design

- Transient streams
- Continuous queries
- Sequential access
- Bounded main memory
- Historical data is important
- Real-time requirements
- Possibly multi-GB arrival rate
- Data at fine granularity
- Data stale/imprecise
- Unpredictable/variable data arrival and characteristics

Ack. From Motwani’s PODS tutorial slides
Architecture: Stream Query Processing

SDMS (Stream Data Management System)

Multiple streams

Continuous Query

User/Application

Results

Stream Query Processor

Scratch Space (Main memory and/or Disk)
Challenges of Stream Data Processing

- Multiple, continuous, rapid, time-varying, ordered streams
- Main memory computations
- Queries are often continuous
  - Evaluated continuously as stream data arrives
  - Answer updated over time
- Queries are often complex
  - Beyond element-at-a-time processing
  - Beyond stream-at-a-time processing
  - Beyond relational queries (scientific, data mining, OLAP)
- Multi-level/multi-dimensional processing and data mining
  - Most stream data are at low-level or multi-dimensional in nature
Processing Stream Queries

- Query types
  - One-time query vs. continuous query (being evaluated continuously as stream continues to arrive)
  - Predefined query vs. ad-hoc query (issued on-line)

- Unbounded memory requirements
  - For real-time response, main memory algorithm should be used
  - Memory requirement is unbounded if one will join future tuples

- Approximate query answering
  - With bounded memory, it is not always possible to produce exact answers
  - High-quality approximate answers are desired
  - Data reduction and synopsis construction methods
    - Sketches, random sampling, histograms, wavelets, etc.
Methodologies for Stream Data Processing

- Major challenges
  - Keep track of a large universe, e.g., pairs of IP address, not ages
- Methodology
  - Synopses (trade-off between accuracy and storage)
  - Use *synopsis data structure*, much smaller ($O(\log^k N)$ space) than their base data set ($O(N)$ space)
  - Compute an *approximate answer* within a *small error range* (factor $\varepsilon$ of the actual answer)
- Major methods
  - Random sampling
  - Histograms
  - Sliding windows
  - Multi-resolution model
  - Sketches
  - Randomized algorithms
Stream Data Processing Methods (1)

- Random sampling (but without knowing the total length in advance)
  - Reservoir sampling: maintain a set of $s$ candidates in the reservoir, which form a true random sample of the element seen so far in the stream. As the data stream flow, every new element has a certain probability ($s/N$) of replacing an old element in the reservoir.

- Sliding windows
  - Make decisions based only on recent data of sliding window size $w$
  - An element arriving at time $t$ expires at time $t + w$

- Histograms
  - Approximate the frequency distribution of element values in a stream
  - Partition data into a set of contiguous buckets
  - Equal-width (equal value range for buckets) vs. V-optimal (minimizing frequency variance within each bucket)

- Multi-resolution models
  - Popular models: balanced binary trees, micro-clusters, and wavelets
Approximate Query Answering in Streams

- Sliding windows
  - Only over sliding windows of *recent stream data*
  - Approximation but often more desirable in applications

- Batched processing, sampling and synopses
  - **Batched** if update is fast but computing is slow
    - Compute periodically, not very timely
  - **Sampling** if update is slow but computing is fast
    - Compute using sample data, but not good for joins, etc.
  - **Synopsis** data structures
    - Maintain a small *synopsis* or *sketch* of data
    - Good for querying historical data

- Blocking operators, e.g., sorting, avg, min, etc.
  - **Blocking** if unable to produce the first output until seeing the entire input
Projects on DSMS (Data Stream Management System)

- Research projects and system prototypes
  - **STREAM** (Stanford): A general-purpose DSMS
  - **Cougar** (Cornell): sensors
  - **Aurora** (Brown/MIT): sensor monitoring, dataflow
  - **Hancock** (AT&T): telecom streams
  - **Niagara** (OGI/Wisconsin): Internet XML databases
  - **OpenCQ** (Georgia Tech): triggers, incr. view maintenance
  - **Tapestry** (Xerox): pub/sub content-based filtering
  - **Telegraph** (Berkeley): adaptive engine for sensors
  - **Tradebot** ([www.tradebot.com](http://www.tradebot.com)): stock tickers & streams
  - **Tribeca** (Bellcore): network monitoring
  - **MAIDS** (UIUC/NCSA): Mining Alarming Incidents in Data Streams
Stream Data Mining vs. Stream Querying

Stream mining—A more challenging task in many cases
- It shares most of the difficulties with stream querying
  - But often requires less “precision”, e.g., no join, grouping, sorting
- Patterns are hidden and more general than querying
- It may require exploratory analysis
  - Not necessarily continuous queries

Stream data mining tasks
- Multi-dimensional on-line analysis of streams
- Mining outliers and unusual patterns in stream data
- Clustering data streams
- Classification of stream data
Concept drift

• In many applications, the distribution underlying the instances or the rules underlying their labeling may change over time, i.e. the goal of the prediction, the class to be predicted or the target value to be predicted, may change over time. This problem is referred to as concept drift.
Episode Rules

• Association rules applied to sequences of events.

• *Episode* – set of event predicates and partial ordering on them
• **Association rules describe how things occur together in the data**
  – E.g., "IF an alarm has certain properties, THEN it will have other given properties"

• **Episode rules describe temporal relationships between things**
  – E.g., "IF a certain combination of alarms occurs within a time period, THEN another combination of alarms will occur within a time period"
Basics

• As defined earlier, telecom data contains alarms:

1234 EL1 PCM 940926082623 A1 ALARMTXT..

  | Alarm type | Date, time | Alarm severity class |
  | Alarming network element |
  | Alarm number |

• Now we forget about relationships between attributes within alarms as with the association rules

• We just take the alarm number attribute, handle it here as event/alarm type and inspect the relationships between events/alarms
Episodes

• Partially ordered set of pages
• *Serial episode* – totally ordered with time constraint
• *Parallel episode* – partial ordered with time constraint
• *General episode* – partial ordered with no time constraint
DAG for Episode
• **Data:**
  – Data is a set $R$ of *events*
  – Every *event* is a pair $(A, t)$, where
    • $A \in R$ is the *event type* (e.g., alarm type)
    • $t$ is an integer, the *occurrence time* of the event
  – *Event sequence* $s$ on $R$ is a triple $(s, T_s, T_e)$
    • $T_s$ is starting time and $T_e$ is ending time
    • $T_s < T_e$ are integers
    • $s = \langle (A_1, t_1), (A_2, t_2), \ldots, (A_n, t_n) \rangle$
    • $A_i \in R$ and $T_s \leq t_i < T_e$ for all $i=1, \ldots, n$
• Example alarm data sequence:

$$\langle (D, 10), (C, 20), \ldots, (A, 150) \rangle$$

• Here:
  – $A$, $B$, $C$ and $D$ are event (or here alarm) types
  – $10...150$ are occurrence times
  – $s = \langle (D, 10), (C, 20), \ldots, (A, 150) \rangle$
  – $T_s$ (starting time) = 10 and $T_e$ (ending time) = 150

• Note: There needs not to be events on every time slot!
**Episodes:**

- An episode is a pair \((V, \leq)\)
  - \(V\) is a collection of event types, e.g., alarm types
  - \(\leq\) is a partial order on \(V\)

- Given a sequence \(S\) of alarms, an episode \(\alpha = (V, \leq)\) occurs within \(S\) if there is a way of satisfying the event types (e.g., alarm types) in \(V\) using the alarms of \(S\) so that the partial order \(\leq\) is respected

- **Intuitively:** episodes consist of alarms that have certain properties and occur in a certain partial order
The most useful partial orders are:

- **Total orders**
  - The predicates of each episode have a fixed order
  - Such episodes are called *serial* (or "ordered")

- **Trivial partial orders**
  - The order of predicates is not considered
  - Such episodes are called *parallel* (or "unordered")

- **Complicated?**
  - Not really, let's take some clarifying examples
• **Examples:**

[Diagram showing three scenarios:]

- **Serial episode:** A → B
- **Parallel episode:** A ↔ B
- **More complex episode with serial and parallel:** A ↔ B ↔ C
WINEPI Approach

• The name of the WINEPI method comes from the technique it uses: a sliding window

• Intuitively:
  – A window is slided through the event-based data sequence
  – Each window "snapshot" is like a row in a database
  – The collection of these "snapshots" forms the rows in the database

• Complicated?
  – Not really, let's take a clarifying example
• Example alarm data sequence:

D C A B D A B C

0 10 20 30 40 50 60 70 80 90

• The window width is 40 seconds, last point excluded
• The first/last window contains only the first/last event
Formally, given a set $E$ of event types an event sequence $S = (s, T_s, T_e)$ is an ordered sequence of events $event_i$ such that $event_i \leq event_{i+1}$ for all $i=1, \ldots, n-1$, and $T_s \leq event_i < T_e$ for all $i=1, \ldots, n$.
Formally, a window on event sequence $S$ is an event sequence $S=(w,t_s,t_e)$, where $t_s < T_e$, $t_e > T_s$, and $w$ consists of those pairs $(event, t)$ from $s$ where $t_s \leq t < t_e$.

The value $t_s \leq t < t_e$ is called window width, $W$. 

Diagram:

- $T_s$
- $t_1$ $t_2$ $t_3$ $t_e$ $t_n$
- $W$
• By definition, the first and the last windows on a sequence extend outside the sequence, so that the last window contains only the first time point of the sequence, and the last window only the last time point.
• The **frequency** (cf. support with association rules) of an episode \( \alpha \) is the fraction of windows in which the episode occurs, i.e.,

\[
fr(\alpha, S, W) = \frac{|S_w \in W(S, W) | \alpha \text{ occurs in } S_w |}{|W(S, W)|}
\]

where \( W(S, W) \) is the set of all windows \( S_w \) of sequence \( S \) such that the window width is \( W \)
• When searching for the episodes, a **frequency threshold** (cf. support threshold with association rules) $min_{fr}$ is used

• Episode $\alpha$ is *frequent* if $fr(\alpha, s, win) \geq min_{fr}$, i.e, "if the frequency of $\alpha$ exceeds the minimum frequency threshold within the data sequence $s$ and with window width $win$"

• $F(s, win, min_{fr})$: a collection of frequent episodes in $s$ with respect to $win$ and $min_{fr}$

• **Apriori trick holds**: if an episode $\alpha$ is frequent in an event sequence $s$, then all subepisodes $\beta \prec \alpha$ are frequent
Formally, an episode rule is an expression \( \beta \Rightarrow \gamma \), where \( \beta \) and \( \gamma \) are episodes such that \( \beta \) is a subepisode of \( \gamma \).

An episode \( \beta \) is a subepisode of \( \gamma \) (\( \beta \prec \gamma \)), if the graph representation \( \beta \) is a subgraph of the representation of \( \gamma \).
WINEPI Approach

• The fraction

\[
\frac{fr(\gamma, S, W)}{fr(\beta, S, W)} = \text{frequency of the whole episode}
\]

\[
\frac{fr(\beta, S, W)}{fr(\beta, S, W)} = \text{frequency of the LHS episode}
\]

is the confidence of the WINEPI episode rule

• The confidence can be interpreted as the conditional probability of the whole of \( \gamma \) occurring in a window, given that \( \beta \) occurs in it
• Intuitively:
  – WINEPI rules are like association rules, but with an additional time aspect:
    If events (alarms) satisfying the rule antecedent (left-hand side) occur in the right order within $W$ time units, then also the rule consequent (right-hand side) occurs in the location described by $\leq$, also within $W$ time units

antecedent $\Rightarrow$ consequent [window width] $(f, c)$
• **Input:** A set $R$ of event/alarms types, an event sequence $s$ over $R$, a set $E$ of episodes, a window width $win$, and a frequency threshold $min_{fr}$

• **Output:** The collection $F(s, win, min_{fr})$

• **Method:**
  1. compute $C_1 := \{ \alpha \in E \mid |\alpha| = 1\}$;
  2. $i := 1$;
  3. **while** $C_i \neq \emptyset$ **do**
  4. (* compute $F(s, win, min_{fr}) := \{ \alpha \in C_i \mid fr(\alpha, s, win) \geq min_{fr}\}$;
  5. $i := i+1$;
  6. (** compute $C_i := \{ \alpha \in E \mid |\alpha| = I, and \beta \in F_{|\beta|}(s, win, min_{fr}) for all \beta \in E, \beta < \alpha\}$;

(* = database pass, (** candidate generation
• First problem: given a sequence and a episode, find out whether the episode occurs in the sequence
• Finding the number of windows containing an occurrence of the episode can be reduced to this
• Successive windows have a lot in common
• How to use this?
  – An incremental algorithm
  – Same idea as for association rules
  – A candidate episode has to be a combination of two episodes of smaller size
  – Parallel episodes, serial episodes
WINEPI Algorithm

- **Parallel episodes:**
  - For each candidate $\alpha$ maintain a counter $\alpha.event\_count$: how many events of $\alpha$ are present in the window
  - When $\alpha.event\_count$ becomes equal to $|\alpha|$, indicating that $\alpha$ is entirely included in the window, save the starting time of the window in $\alpha.inwindow$
  - When $\alpha.event\_count$ decreases again, increase the field $\alpha.freq\_count$ by the number of windows where $\alpha$ remained entirely in the window

- **Serial episodes: use a state automata**
WINEPI Approach

- Example alarm data sequence:

- The window width is 40 secs, movement step 10 secs
- The length of the sequence is 70 secs (10-80)
• By sliding the window, we'll get 11 windows ($U_1$-$U_{11}$):

WINEPI Approach

• Frequency threshold is set to 40%, i.e., an episode has to occur at least in 5 of the 11 windows
<table>
<thead>
<tr>
<th>Window $U_i$</th>
<th>Contents of $U_i$</th>
<th>Parallel episodes occurring in $U_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_1,[-20,20]$</td>
<td>[→, →, →, D]</td>
<td>{D}</td>
</tr>
<tr>
<td>$U_2,[-10,30]$</td>
<td>[→, →, D, C]</td>
<td>{C, D}, {CD}</td>
</tr>
<tr>
<td>$U_3,[0,40]$</td>
<td>[→, D, C, A]</td>
<td>{A, C, D}, {AC, AD, CD}, {ACD}</td>
</tr>
<tr>
<td>$U_4,[10,50]$</td>
<td>[D, C, A, B]</td>
<td>{A, B, C, D}, {AB, AC, AD, BC, BD, CD}, {ABC, ABD, ACD, BCD}, {ABCD}</td>
</tr>
<tr>
<td>$U_5,[20,60]$</td>
<td>[C, A, B, D]</td>
<td>{A, B, C, D}, {AB, AC, AD, BC, BD, CD}, {ABC, ABD, ACD, BCD}, {ABCD}</td>
</tr>
<tr>
<td>$U_6,[30,70]$</td>
<td>[A, B, D, A]</td>
<td>{A, B, D}, {AB, AD, BD}, {ABD}</td>
</tr>
<tr>
<td>$U_7,[40,80]$</td>
<td>[B, D, A, B]</td>
<td>{A, B, D}, {AB, AD, BD}, {ABD}</td>
</tr>
<tr>
<td>$U_8,[50,90]$</td>
<td>[D, A, B, C]</td>
<td>{A, B, C, D}, {AB, AC, AD, BC, BD, CD}, {ABC, ABD, ACD, BCD}, {ABCD}</td>
</tr>
<tr>
<td>$U_9,[60,100]$</td>
<td>[A, B, C, -]</td>
<td>{A, B, C}, {AB, AC, BC}, {ABC}</td>
</tr>
<tr>
<td>$U_{10},[70,110]$</td>
<td>[B, C, -]</td>
<td>{B, C}, {BC}</td>
</tr>
<tr>
<td>$U_{11},[80,120]$</td>
<td>[C, - , -]</td>
<td>{C}</td>
</tr>
</tbody>
</table>
Suppose that the task is to find all parallel episodes:

- First, create singletons, i.e., parallel episodes of size 1 \((A, B, C, D)\)
- Then, recognize the frequent singletons (here all are)
- From those frequent episodes, build candidate episodes of size 2: \(AB, AC, AD, BC, BD, CD\)
- Then, recognize the frequent parallel episodes (here all are)
- From those frequent episodes, build candidate episodes of size 3: \(ABC, ABD, ACD, BCD\)
- When recognizing the frequent episodes, only ABD occurs in more than four windows
- There are no candidate episodes of size four
WINEPI Approach

- Episode frequencies and example rules with WINEPI:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>73%</td>
</tr>
<tr>
<td>C</td>
<td>73%</td>
</tr>
<tr>
<td>A</td>
<td>64%</td>
</tr>
<tr>
<td>B</td>
<td>64%</td>
</tr>
<tr>
<td>D C</td>
<td>45%</td>
</tr>
<tr>
<td>D A</td>
<td>55%</td>
</tr>
<tr>
<td>D B</td>
<td>45%</td>
</tr>
<tr>
<td>C A</td>
<td>45%</td>
</tr>
<tr>
<td>C B</td>
<td>45%</td>
</tr>
<tr>
<td>A B</td>
<td>45%</td>
</tr>
<tr>
<td>D A B</td>
<td>45%</td>
</tr>
</tbody>
</table>

  - D ⇒ A [40] (55%, 75%)
  - D A ⇒ B [40] (45%, 82%)
• **Data:**
  - Alarms from a telecommunication network
  - 73,000 events (7 weeks), 287 event types
  - Parallel and serial episodes
  - Window widths ($W$) 10-120 seconds
  - Window movement = $W/10$
  - $\text{min}_\text{fr} = 0.003$ (0.3%), frequent: about 100 occurrences
  - 90 MHz Pentium, 32MB memory, Linux operating system. The data resided in a 3.0 MB flat text file
## WINEPI: Experimental Results

<table>
<thead>
<tr>
<th>Window width (s)</th>
<th>Serial episodes</th>
<th></th>
<th>Parallel episodes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#frequent</td>
<td>time (s)</td>
<td>#frequent</td>
<td>time (s)</td>
</tr>
<tr>
<td>10</td>
<td>16</td>
<td>31</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>20</td>
<td>31</td>
<td>63</td>
<td>17</td>
<td>9</td>
</tr>
<tr>
<td>40</td>
<td>57</td>
<td>117</td>
<td>33</td>
<td>14</td>
</tr>
<tr>
<td>60</td>
<td>87</td>
<td>186</td>
<td>56</td>
<td>15</td>
</tr>
<tr>
<td>80</td>
<td>145</td>
<td>271</td>
<td>95</td>
<td>21</td>
</tr>
<tr>
<td>100</td>
<td>245</td>
<td>372</td>
<td>139</td>
<td>21</td>
</tr>
<tr>
<td>120</td>
<td>359</td>
<td>478</td>
<td>189</td>
<td>22</td>
</tr>
</tbody>
</table>
• **One shortcoming in WINEPI approach:**
  - Consider that two alarms of type A and one alarm of type B occur in a window
  - Does the parallel episode consisting of A and B appear once or twice?
  - If once, then with which alarm of type A?
• **Alternative approach to discovery of episodes**
  – No sliding windows
  – For each potentially interesting episode, find out the exact occurrences of the episode

• **Advantages:** easy to modify time limits, several time limits for one rule:
  "If A and B occur within 15 seconds, then C follows within 30 seconds"

• **Disadvantages:** uses a lot of space
MINEPI Approach

• Formally, given a episode \( \alpha \) and an event sequence \( S \), the interval \([t_s, t_e]\) is a **minimal occurrence** \( \alpha \) of \( S \),
  – If \( \alpha \) occurs in the window corresponding to the interval
  – If \( \alpha \) does not occur in any proper subinterval

• The **set of minimal occurrences** of an episode \( \alpha \) in a given event sequence is denoted by \( mo(\alpha) \):

\[
mo(\alpha) = \{ [t_s, t_e] | [t_s, t_e] \text{ is a minimal occurrence of } \alpha \}
\]
• Example: Parallel episode $\beta$ consisting of event types $A$ and $B$ has three minimal occurrences in $s$: $\{[30,40], [40,60], [60,70]\}$, $\alpha$ has one occurrence in $s$: $\{[60,80]\}$
• **Informally**, a MINEPI episode rule gives the conditional probability that a certain combination of events (alarms) occurs within some time bound, given that another combination of events (alarms) has occurred within a time bound.

• Formally, an **episode rule** is $\beta [\text{win}_1] \Rightarrow \alpha [\text{win}_2]$

• $\beta$ and $\alpha$ are episodes such that $\beta \prec \alpha$ ($\beta$ is a subepisode of $\alpha$)

• If episode $\beta$ has a minimal occurrence at interval $[t_s, t_e]$ with $t_e - t_s \leq \text{win}_1$, then episode $\alpha$ occurs at interval $[t_s, t'_e]$ for some $t'_e$ such that $t'_e - t_s \leq \text{win}_2$
Pattern discovery

- Pattern = episode rule
- Frequency in sliding windows, or nr of occurrences
- Count – generate candidates, evaluate ...
Pattern Discovery

1. Choose the **language** (formalism) to represent the patterns (search space)

2. Choose the **rating** for patterns, to tell which is “better” than others

3. Design an **algorithm** that **finds the best patterns** from the pattern class, **fast.**

TGTTCTTTTCTTCTTTCATACATCCTTTTTTCCTTTTTTTCC
TTCTCCTTTTCTTTCTTGACTTTTTAATATAAGGCTTACCA
TCCTTTCTTCTTCTTCAATAACCTTCTTCATTGCTTCTTC
TTCGATTGCTTCAAAGTAGGTTCGTGAATCATCCTTCAAT
GCCTCAGCACCTTCAGCACATTGCACATTTCATTCCATTGAA
GTGCTGCACCTGCGCTGTCTTGCTAATGGATTTGGAGTT
GGCGTGGGACTGATTTCTTCTGACATGGGCGGCGCTTCTCT
TCGAATTCCATCAGTCCTCATAGTTCTGTTGGTTCTTTTT
CTCTGATGATCGTCATCTTTCACTGATCTGATGTTCCTG
TGCCCTATCTATATCATCTCATAAGTTCCACCTTTGCCACT
TTCCAAGATCTCTCATTCAAATGGGCTTTAAAGCCGTAC
TTTTTTTCACTCGATGAGCTATAAGAGTTTTCCACTTTTA
GATCGTGGCTGGGCTTATATTACGGTGTGATGAGGGCGC
TTGAAAAGATTTTTTCATCTCACAAGGGAGCGAGGGGCGC
TTGAAAAGATTTTTTTCTCATTCAACAGCGACGAGGGGC
AGTGGTTTGAAGCTAGATGCAGTAGGTGCAAGCGTAGAGT
CTTAGAAGATAAAGTAGTGAATTACATAGATTTGCGTATAC
Patterns: AT
Cluster of co-expressed genes, pattern discovery in regulatory regions

Expression profiles

Retrieval

600 basepairs

Find patterns over-represented within cluster

Brazma et al: Genome Research 1998;
Background - ALL upstream sequences

Cluster: \( \pi \) occurs 3 times

\( P(3,6,0.2) \) is probability of having \( \geq 3 \) matches in 6 sequences

5 out of 25, \( p = 0.2 \)

\( P(\pi,3,6,0.2) = 0.0989 \)
Eukaryotic genome can be thought of as six Levels of DNA structure.

The loops at Level 4 range from 0.5kb to 100kb in length.

If these loops were stabilized then the genes inside the loop would not be expressed.
DNA determines function (?)

DNA  
GenBank / EMBL Bank

Protein  
SwissProt/TrEMBL

Structure  
PDB/Molecular Structure Database

4 Nucleotides  20+ Amino Acids  
(3nt  AA)
A Simple Gene

A:  

B:  

C:  

DNA:  

ATCGAAAT  

TAGCTTTA  

+Modifications
YGR128C + 100

101 Sequences relative to ORF start

GATGAG.T     1:52/70  2:453/508  R:7.52345  BP:1.02391e-33
AAAATTTT     1:63/77  2:833/911  R:4.95687  BP:5.02807e-32
TGAAAA.TTT   1:45/53  2:333/350  R:8.85687  BP:1.69905e-31
TG.AAA.TTTT  1:40/43  2:254/260  R:10.3214   BP:3.84624e-30
TGAAA..TTT   1:54/65  2:608/645  R:5.82106   BP:1.0887e-29

...
Sequence patterns: the basis of the SPEXS
SPEXS: general algorithm

1. $S = \text{input sequences (} ||S||=n \text{) }$
2. $e = \text{empty pattern, } e.\text{pos} = \{1,\ldots,n\}$
3. enqueue( order , e )

4. while $p = \text{dequeue( order )}$
   5. generate all allowed extensions $p'$ of $p$ (& $p'.\text{pos}$)
   6. enqueue( order, $p'$, priority($p'$) )
   7. enqueue( output, $p'$, fitness($p'$) )

8. while $p = \text{dequeue( output )}$
9. Output p

Jaak Vilo: Discovering Frequent Patterns from Strings.
Technical Report C-1998-9 (pp. 20) May 1998. Department of Computer Science,
University of Helsinki.

Jaak Vilo: Pattern Discovery from Biosequences
PhD Thesis, Department of Computer Science, University of Helsinki, Finland.
Report A-2002-3 Helsinki, November 2002, 149 pages

Applications in bioinformatics:
- Functional elements in proteins (2002: 32 cit)
.G.GATGAG.T. 39 seq

39 seq (vs 193)
p= 2.5e-33
-1: .G.GATGAG.T. 61 seq (vs 1292)
p = 1.4e-19
-2: .G.GATGAG.T. 91 seq (vs 5464)
-3: G.GATGAG.T. 98 seq
<table>
<thead>
<tr>
<th>Pattern</th>
<th>In cluster</th>
<th>Total nr</th>
<th>Ratio</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>G.GATGAG.T</td>
<td>39</td>
<td>193</td>
<td>13.24</td>
<td>2.490e-33</td>
</tr>
<tr>
<td>TG.AAA.TTT</td>
<td>53</td>
<td>538</td>
<td>6.46</td>
<td>3.248e-31</td>
</tr>
<tr>
<td>TGAAAAA.TTT</td>
<td>45</td>
<td>333</td>
<td>8.86</td>
<td>1.699e-31</td>
</tr>
<tr>
<td>-1:G.GATGAG.T</td>
<td>61</td>
<td>1295</td>
<td>3.09</td>
<td>1.441e-19</td>
</tr>
<tr>
<td>-1:TG.AAA.TTT</td>
<td>89</td>
<td>3836</td>
<td>1.52</td>
<td>6.126e-12</td>
</tr>
<tr>
<td>-1:TGAAAAA.TTT</td>
<td>76</td>
<td>2190</td>
<td>2.27</td>
<td>1.654e-18</td>
</tr>
</tbody>
</table>

![Diagram of sequences with annotations]

Jaak Vilo: Pattern Discovery from Biosequences
PhD Thesis, Department of Computer Science, University of Helsinki, Finland
These hits result in a PWM:
PWM based on all previous hits, here shown highest-scoring occurrences in blue.
GPCR coupling

Signal: Agonist

Current perspective

GPCR:

G-protein

Effector enzymes/channels

Intracellular messengers
GPCRs

• Sense and transduce a huge variety of:
  • hormones, neurotransmitters, peptides, photons, tastants, odourants, ions, peptidases, and pheromones

• Involved in practically all aspects of human physiology, and also pathophysiology

• Estimates of 800-2000 members of the superfamily in the human genome
Drugs Acting Upon GPCRs

• β-Adrenoceptor agonists/antagonists
  • Asthma, high blood pressure, anxiety
• L-DOPA (dopamine precursor)
  • Agent of choice for Parkinson’s disease
• Histamine H1 antagonists
  • Allergic and anaphylactic reactions, travel sickness
• Histamine H2 antagonists
  • Ulcer treatment
• Opioid agonists
  • Powerful analgesics
• 5-HT$_{1D}$ antagonists
  • Migraine
Diseases Associated with GPCRs

- Colour blindness
  - *Cone opsin receptor*
- Hyperthyroidism
  - *Thyrotropin receptor*
- X-linked diabetes insipidus
  - *V2 vasopressin receptor*
Receptor-G Protein Coupling

- We know:
  - 20 G protein $\alpha$ subunits, these determine the functional class of the G protein heterotrimer. By sequence similarity these can be split into 4 groups.
  - From a functional perspective there are three well-recognised classes of G proteins
    - $G_s$ stimulate adenylate cyclase (includes $G_{olf}$)
    - $G_{i/o}$ inhibit adenylate cyclase (includes $G_t, G_z$)
    - $G_{q/11}$ stimulate PI hydrolysis
    - $G_{a12/13}$ unknown function
  - No easy way to predict from GPCR its coupling specificity to $G_s, G_{i/o}, G_{q/11}$
  - GPCRs may exhibit selective, or promiscuous coupling
Computational Approaches

Structural Analysis?

We have only one crystal structure -1f88.
Our Computational Approach

- Using a new membrane topology prediction algorithm (designed specifically for GPCRs), we constrained our pattern search to the intracellular domains of ≈ 100 receptor sequences with well-characterised, and non-promiscuous coupling (split into $G_s$, $G_{i/o}$ and $G_{q/11}$).
Receptor Match Positions

Croning, Vilo, Möller, ISMB 2001
Phylogenetic Tree?

Receptors
- $G_{i/o}$
- $G_{q/11}$
- $G_s$
More Topics

• Information Retrieval
• Text Mining
• Web Mining
• Social Network Analysis
  – friends, epidemiology, co-authoring, co-citation, espionage, ...
• Graph Mining
Concepts

*Information Retrieval* - the study of systems for representing, indexing (organising), searching (retrieving), and recalling (delivering) data.

*Information Filtering* - given a large amount of data, return the data that the user wants to see

*Information Need* - what the user really wants to know; a query is an approximation to the information need.

*Query* - a string of words that characterizes the information that the user seeks

*Browsing* - a sequence of user interaction tasks that characterizes the information that the user seeks
Documents

- News, articles
- Laws, legal documents
- Scientific publications, patents
- E-mail
- Technical documents
- Books
- Encyclopedias
- Dictionaries
- ...
Information Retrieval

• DB of indexed documents

• Query

• Find documents relevant to query
The classic search model

TASK

Info Need

Verbal form

Mis-conception

Mis-translation

Mis-formulation

Query

SEARCH ENGINE

Query Refinement

Results

Corpus

Find this: mouse trap any language

Polysemy Synonymy

Get rid of mice in a politically correct way

Info about removing mice without killing them

How do I trap mice alive?
More features

- Metadata
- Context
- Hypertext – xrefs
- Language
- Structured vs unstructured
- Semantics
- Tags
- ...

Jaak Vilo and other authors
Classic IR Goal

- Classic relevance
  - For each query $Q$ and stored document $D$ in a given corpus assume there exists relevance $\text{Score}(Q, D)$
    - Score is average over users $U$ and contexts $C$
  - Optimize $\text{Score}(Q, D)$ as opposed to $\text{Score}(Q, D, U, C)$
  - That is, usually:
    - Context ignored
    - Individuals ignored
    - Corpus predetermined

Bad assumptions in the web context
The Notion of Relevance

- Data retrieval: semantics tied to syntax
- Information retrieval: ambiguous semantics
- Relevance:
  - Depends on the user
  - Depends on the context (task, time, etc)
  - Corollary: The Perfect IR System does not exist
Evaluation: First Quality, next Efficiency

Database

Answer

Relevant Documents

TP
FP
FN
TN

T=True
F=False
P=Positives
N=Negative

Precision = \frac{\text{Answer}}{\text{Answer}}

Recall = \frac{\text{Rel. Docs}}{\text{Rel. Docs}}

An introduction
User Needs

- **Need** (Broder 2002)
  - **Informational** – want to learn about something (~40% / 65%)
    - Low hemoglobin
  - **Navigational** – want to go to that page (~25% / 15%)
    - United Airlines
  - **Transactional** – want to do something (web-mediated) (~35% / 20%)
    - Access a service
    - Downloads
    - Shop
    - Edinburgh weather
    - Mars surface images
    - Canon S410
  - Gray areas
    - Find a good hub
    - Exploratory search “see what’s there”
    - Car rental Brasil
Example: *I want information about Santiago*

**Context**
- Family in Chile
- Catholic
- Travelling to Cuba
- Lives in Argentina
- Located in Santo Domingo
- Architect
- Spanish movies fan
- Baseball fan

**Probable Answer**
- Santiago de Chile
- Santiago de Compostela
- Santiago de Cuba
- Santiago del Estero
- Santiago de los Caballeros
- Santiago Calatrava
- Santiago Segura
- Santiago Benito
<table>
<thead>
<tr>
<th>SEARCH GOAL</th>
<th>DESCRIPTION</th>
<th>EXAMPLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigational</td>
<td>My goal is to go to specific known website that I already have in mind. The only reason I'm searching is that it's more convenient than typing the URL, or perhaps I don't know the URL.</td>
<td>aloha airlines, duke university hospital, kelly blue book</td>
</tr>
<tr>
<td>2. Informational</td>
<td>My goal is to learn something by reading or viewing web pages</td>
<td>what is a supercharger, 2004 election dates, baseball death and injury, why are metals shiny</td>
</tr>
<tr>
<td>2.1 Directed</td>
<td>I want to learn something in particular about my topic</td>
<td>color blindness, jfk jr</td>
</tr>
<tr>
<td>2.1.1 Closed</td>
<td>I want to get an answer to a question that has a single, unambiguous answer.</td>
<td>help quitting smoking, walking with weights</td>
</tr>
<tr>
<td>2.1.2 Open</td>
<td>I want to get an answer to an open-ended question, or one with unconstrained depth.</td>
<td>pella windows, phone card, travel</td>
</tr>
<tr>
<td>2.2 Undirected</td>
<td>I want to learn anything/everything about my topic. A query for topic X might be interpreted as &quot;tell me about X.&quot;</td>
<td>amsterdam universities, florida newspapers</td>
</tr>
<tr>
<td>2.3 Advice</td>
<td>I want to get advice, ideas, suggestions, or instructions.</td>
<td>kaza lite, mame roms, xxx porn, movie free, live camera in l.a., weather, measure converter</td>
</tr>
<tr>
<td>2.4 Locate</td>
<td>My goal is to find out whether/where some real world service or product can be obtained</td>
<td>tv, cable, phone card</td>
</tr>
<tr>
<td>2.5 List</td>
<td>My goal is to get a list of plausible suggested websites (i.e. the search result list itself), each of which might be candidates for helping me achieve some underlying, unspecified goal.</td>
<td>amsterdam universities, florida newspapers</td>
</tr>
<tr>
<td>3. Resource</td>
<td>My goal is to obtain a resource (not information) available on web pages</td>
<td>free jack o lantern patterns, ellis island lesson plans, house document no. 587</td>
</tr>
<tr>
<td>3.1 Download</td>
<td>My goal is to download a resource that must be on my computer or other device to be useful.</td>
<td>kazaa lite, mame roms, xxx porn, movie free, live camera in l.a., weather, measure converter</td>
</tr>
<tr>
<td>3.2 Entertainment</td>
<td>My goal is to be entertained simply by viewing items available on the result page</td>
<td>free jack o lantern patterns, ellis island lesson plans, house document no. 587</td>
</tr>
<tr>
<td>3.3 Interact</td>
<td>My goal is to interact with a resource using another program/service available on the web site I find.</td>
<td>free jack o lantern patterns, ellis island lesson plans, house document no. 587</td>
</tr>
<tr>
<td>3.4 Obtain</td>
<td>My goal is to obtain a resource that does not require a computer to use. I may print it out, but I can also just look at it on the screen. I'm not obtaining it to learn some information, but because I want to use the resource itself.</td>
<td>free jack o lantern patterns, ellis island lesson plans, house document no. 587</td>
</tr>
</tbody>
</table>
Web search

- data mining university of tartu fall 2009
- Java
- Paris Hilton
- climate change
Global Warming Hoax
links to headlines about global warming and the global warming hoax.
www.doricalorraine.com/ - Cached - Similar - ● ▲ ▾ ▽ ▼

The great 'global warming' hoax
23 Nov 2009 ... While all eyes focus on the unfolding drama of the health care reform/health insurance reform/jobs bill, another critical part of the Change ...
www.wnd.com/index.php?pageid=116682 - Cached - ● ▲ ▾ ▽ ▼

"Climategate" exposes global warming hoax» Evansville Courier & Press
6 Dec 2009 ... The hoax of man-made global warming is being exposed.
www.courierpress.com/.../quotclimategatequot-exposes-global-warming-hoax/ - ● ▲ ▾ ▽ ▼

Moonbattery Blowing Holes in the Global Warming Hoax
21 Jan 2009 ... Blowing Holes in the Global Warming Hoax. You might want to print out articles like this one, in case they start disappearing under an odict ...
www.moonbattery.com/archives/2009/.../blowing_holes_i.html - Cached - Similar - ● ▲ ▾ ▽ ▼

Certainty of Catastrophic Global Warming is a Hoax by James K...
Yes, the Earth’s surface has warmed a bit over the past century, but is that warming caused mainly by humans or by natural cycles?
www.capmag.com/article.asp?ID=5400 - Cached - Similar - ● ▲ ▾ ▽ ▼

ROFASix: Global Warming Hoax Revealed
"Global Warming. It is the hoax. It is bad science. It is a high jacking of public policy. It is no joke. It is the greatest scam in history " ...
rofasix.blogspot.com/.../global-warming-hoax-revealed.html - Cached - Similar - ● ▲ ▾ ▽ ▼

Global Warming Hoax Conclusion - Humor: Fueled by Petrol
25 Nov 2009 ... Jimmy is on his way to Greenland to confront a leading Global Warming Hoaxster...see "Global Warming...Hoax?" from last Wed.). ...
tucsoncitizen.com/petrol/2009/11/global-warming-hoax-conclusion/ - Cached - ● ▲ ▾ ▽ ▼

How Convenient! Govt Destroys Global Warming Hoax Data "suspicious ...
11 Oct 2009 ... This is rich, really rich. What a laugh, and the joke is on us. Big time. Any Republican who signs off on the bankrupting of America is a ...
“Do no evil”

Corporate bias?
Social Mining (2002)

Graphs showing search trends over time:
- Iraq
- Eminem, Jennifer Lopez, Shakira, David Beckham, Ronaldo
- Spain, Italy, Germany, USA, UK
What does WWW look like?
**Fig. 7.** The bowtie structure of the Web: Plot (a) shows the 4 parts: IN, OUT, SCC, and TENDRILS [Broder et al. 2000]. Plot (b) shows recursive bowties: subgraphs of the WWW can each be considered a bowtie. All these smaller bowties are connected by the navigational backbone of the main SCC of the Web [Dill et al. 2001].
Fig. 6. The Internet as a Jellyfish. The Internet AS-level graph can be thought of as a core, surrounded by concentric layers around the core. There are many one-degree nodes that hang off the core and each of the layers.
PageRank

- [Brin and Page, 1998]
- Algorithm suggested $\alpha$ for ranking results in web search
- An authority score is assigned to each Web page
- Authority scores independent of the query

Authority scores correspond to the stationary distribution of a random walk on the graph:
- With probability $\alpha$, follow a link in the graph
- With probability $1-\alpha$, go to a node chosen uniformly at random (teleportation)

- Random walk also known as random surfer model

An introduction to Web Mining, WWW2008, Beijing
Let $E$ be the adjacency matrix of the graph, and $L$ the row-stochastic version of $E$.

Each row of $E$ is normalized so that it sums to 1.

Authority score defined by

$$p_{(i+1)} = L^T p_{(i)}$$

problematic if the graph is not strongly connected, So:

$$p_{(i+1)} = \alpha L^T p_{(i)} + (1-\alpha) \frac{1}{n} 1$$

where $1$ is the matrix with all entries equal to 1

and $\alpha \in [0,1]$, common value $\alpha = 0.85$
Topics

• Information Retrieval
• Text Mining
• Web Mining
• Social Network Analysis
  – friends, epidemiology, co-authoring, co-citation, espionage, ...
• Graph Mining
Twitter Social Network, 20K nodes 250K edges

Image Copyright UMBC eBiquity Research Group
Skype Fast Facts

- +170 million monthly connected users
  - 30 million concurrent users
- +300 million minutes of Skype video/day
  - 42% of all calls include video
  - Goal: reach 1 billion users daily
- Context: 2,1 billion Internet users
Figure XI.2. Connections between the 9/11 hijackers.

the drawing. Many of the “higher level” aesthetic criteria are implicit consequences of the

- minimized number of edge crossings,
- evenly distributed edge length,
- evenly distributed vertex positions on the graph area,
- sufficiently large vertex-edge distances,
- sufficiently large angular resolution between edges.
## Hijackes by Flight

<table>
<thead>
<tr>
<th>Flight 77 : Pentagon</th>
<th>Flight 11 : WTC 1</th>
<th>Flight 175 : WTC 2</th>
<th>Flight 93: PA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Khalid Al-Midhar</td>
<td>Satam Al Suqami</td>
<td>Marwan Al-Shehhi</td>
<td>Saeed Alghamdi</td>
</tr>
<tr>
<td>Majed Moqed</td>
<td>Waleed M. Alshehri</td>
<td>Fayez Ahmed</td>
<td>Ahmed Alhaznawi</td>
</tr>
<tr>
<td>Nawaq Alhamzi</td>
<td>Wail Alshehri</td>
<td>Ahmed Alghamdi</td>
<td>Ahmed Alnami</td>
</tr>
<tr>
<td>Salem Alhamzi</td>
<td>Mohamed Atta</td>
<td>Hamza Alghamdi</td>
<td>Ziad Jarrahi</td>
</tr>
<tr>
<td>Hani Hanjour</td>
<td>Abdulaziz Alomari</td>
<td>Mohald Alshehri</td>
<td></td>
</tr>
</tbody>
</table>
a spring that pulls the vertices together), whereas distinct vertices are pushed apart by some constraint to help prevent them from being drawn at the same point. The method seeks equilibrium of these contradicting constraints. The first such algorithm was introduced by Eades (Eades 1984). Following Eades, two additional layout algorithms were introduced by Kamada and Kawai (KK) (Kamada and Kawai 1989) and Fruchterman and Reingold (FR) (Fruchterman and Reingold 1991).

**Kamada and Kawai's (KK) Method**

Utilizing Hooke’s law, Kamada and Kawai modeled a graph as a system of springs. Every two vertices are connected by a spring whose rest length is proportional to the graph-theoretic distance between its two endpoints. Each spring’s stiffness is inversely proportional to the square of its rest length. The optimization algorithm
Figure XI.7. Summary diagram of centrality measures (solid arrows point to highest value; dashed arrows point to second largest (done using Netminer (Cyram 2004))).
258  

**Link Analysis**

![Graph](image)

**Figure XI.8.** Core partitioning of the hijackers’ graph.

**XI.5.1 Cores**
Semantic search: “Most popular drink that is available on bars that are visited by my friends"
Semantic search: “Most popular drink that is available on bars that are visited by my friends"
Semantic search: “Most popular drink that is available on bars that are visited by my friends"
Planetary-Scale Views on a Large Instant-Messaging Network

Jure Leskovec*
Carnegie Mellon University
jure@cs.cmu.edu

Eric Horvitz
Microsoft Research
horvitz@microsoft.com

ABSTRACT

We present a study of anonymized data capturing a month of high-level communication activities within the whole of the Microsoft Messenger instant-messaging system. We examine characteristics and patterns that emerge from the collective dynamics of large numbers of people, rather than the actions and characteristics of individuals. The dataset contains summary properties of 30 billion conversations among 240 million people. From the data, we construct a communication graph with 180 million nodes and 1.3 billion undirected edges, creating the largest social network constructed and analyzed to date. We report on multiple aspects of the dataset and synthesized graph. We find that the graph is well-connected and robust to node removal. We investigate on a planetary-scale the oft-cited report that people are separated by “six degrees of separation” and find that the average path length among Messenger users is 6.6. We also find that people tend to communicate more with each other when they have similar age, language, and location, and that cross-gender conversations are both more frequent and of longer duration than conversations with the same gender.

Categories and Subject Descriptors: H.2.8 Database Management: : Database applications – Data mining

General Terms: Measurement; Experimentation.

Keywords: Social networks; Communication networks; User demographics; Large data; Online communication.
Figure 4: World and Messenger user population age pyramid. Ages 15–30 are overrepresented in the Messenger population.
Figure 2: (a) Distribution of the number of people participating in a conversation. (b) Distribution of the durations of conversations. The spread of durations can be described by a power-law distribution.

Figure 3: (a) Distribution of login duration. (b) Duration of times when people are not logged into the system (times between logout and login).
Figure 6: Communication characteristics of users by reported age. We plot age vs. age and the color (z-axis) represents the intensity of communication.
Figure 7: Number of users at a particular geographic location. Color of dots represents the number of users.

Figure 8: Number of Messenger users per capita. Color intensity corresponds to the number of users per capita in the cell of the grid.
Figure 9: A communication heat map.

Figure 10: (a) Communication among countries with at least 10 million conversations in June 2006. (b) Countries by average length of the conversation. Edge widths correspond to logarithms of intensity of links.
Nov 19, 2011

- Facebook has released two papers about their network:
- The Anatomy of the Facebook Social Graph
- Four Degrees of Separation
Figure 2: The probability mass functions of the distance distributions of the current graphs (truncated at distance 10).

Figure 3: The average distance graph. See also Table 6.
Figure 6. Degree correlations. (a) The average neighbor degree of an individual with degree $k$ is the solid line. The horizontal dashed line shows the expected value if there were no degree correlations in the network $\langle k'^2 \rangle / \langle k \rangle$, and the diagonal is shown as a dashed line. (b) The conditional probability $p(k'|k)$ that a randomly chosen neighbor of an individual with degree $k$ has degree $k'$. The solid lines, on the linear-log scale, show the measured values for four distinct degrees $k$ shown in the caption. The orange line shows the expected distribution, $\frac{k'^2}{\langle k \rangle}$, if the degrees were uncorrelated.
Figure 8. The distribution $p(t'|t)$ of ages $t'$ for the neighbors of users with age $t$. The solid lines show the measured distributions against the age $t$ described in the legend, and the red line shows the distribution of ages found by following a randomly chosen edge in the network.
Figure 9. Normalized country adjacency matrix. Matrix of edges between countries with > 1 million users and > 50% Facebook penetration shown on a log scale. To normalize, we divided each element of the adjacency matrix by the product of the row country degree and column country degree.
<table>
<thead>
<tr>
<th>Country</th>
<th>ISO code</th>
<th>Country</th>
<th>ISO code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indonesia</td>
<td>ID</td>
<td>United Kingdom</td>
<td>GB</td>
</tr>
<tr>
<td>Philippines</td>
<td>PH</td>
<td>South Africa</td>
<td>ZA</td>
</tr>
<tr>
<td>Sri Lanka</td>
<td>LK</td>
<td>Israel</td>
<td>IL</td>
</tr>
<tr>
<td>Australia</td>
<td>AU</td>
<td>Jordan</td>
<td>JO</td>
</tr>
<tr>
<td>New Zealand</td>
<td>NZ</td>
<td>United Arab Emirates</td>
<td>AE</td>
</tr>
<tr>
<td>Thailand</td>
<td>TH</td>
<td>Kuwait</td>
<td>KW</td>
</tr>
<tr>
<td>Malaysia</td>
<td>MY</td>
<td>Algeria</td>
<td>DZ</td>
</tr>
<tr>
<td>Singapore</td>
<td>SG</td>
<td>Tunisia</td>
<td>TN</td>
</tr>
<tr>
<td>Hong Kong</td>
<td>HK</td>
<td>Italy</td>
<td>IT</td>
</tr>
<tr>
<td>Taiwan</td>
<td>TW</td>
<td>Macedonia</td>
<td>MK</td>
</tr>
<tr>
<td>United States</td>
<td>US</td>
<td>Albania</td>
<td>AL</td>
</tr>
<tr>
<td>Dominican Republic</td>
<td>DO</td>
<td>Serbia</td>
<td>RS</td>
</tr>
<tr>
<td>Puerto Rico</td>
<td>PR</td>
<td>Slovenia</td>
<td>SI</td>
</tr>
<tr>
<td>Mexico</td>
<td>MX</td>
<td>Bosnia and Herzegovina</td>
<td>BA</td>
</tr>
<tr>
<td>Canada</td>
<td>CA</td>
<td>Croatia</td>
<td>HR</td>
</tr>
<tr>
<td>Venezuela</td>
<td>VE</td>
<td>Turkey</td>
<td>TR</td>
</tr>
<tr>
<td>Chile</td>
<td>CL</td>
<td>Portugal</td>
<td>PT</td>
</tr>
<tr>
<td>Argentina</td>
<td>AR</td>
<td>Belgium</td>
<td>BE</td>
</tr>
<tr>
<td>Uruguay</td>
<td>UY</td>
<td>France</td>
<td>FR</td>
</tr>
<tr>
<td>Colombia</td>
<td>CO</td>
<td>Hungary</td>
<td>HU</td>
</tr>
<tr>
<td>Costa Rica</td>
<td>CR</td>
<td>Ireland</td>
<td>IE</td>
</tr>
<tr>
<td>Guatemala</td>
<td>GT</td>
<td>Denmark</td>
<td>DK</td>
</tr>
<tr>
<td>Ecuador</td>
<td>EC</td>
<td>Norway</td>
<td>NO</td>
</tr>
<tr>
<td>Peru</td>
<td>PE</td>
<td>Sweden</td>
<td>SE</td>
</tr>
<tr>
<td>Bolivia</td>
<td>BO</td>
<td>Czech Republic</td>
<td>CZ</td>
</tr>
<tr>
<td>Spain</td>
<td>ES</td>
<td>Bulgaria</td>
<td>BG</td>
</tr>
<tr>
<td>Ghana</td>
<td>GH</td>
<td>Greece</td>
<td>GR</td>
</tr>
</tbody>
</table>

Table 1. ISO country codes used as labels in Figure 9.
4.1 Setup

The computations were performed on a 24-core machine with 72 GiB of memory and 1 TiB of disk space. The first task was to import the Facebook graph(s) into a compressed form for WebGraph [4], so that the multiple scans required by HyperANF’s diffusive process could be carried out relatively quickly. This part required some massaging of Facebook’s internal IDs into a contiguous numbering: the resulting current fb graph (the largest we analysed) was compressed to 345 GB at 20 bits per arc, which is 86% of the information-theoretical lower bound ($\log{\binom{n^2}{m}}$ bits, there $n$ is the number of nodes and $m$ the number of arcs). Whichever coding we

---

5To establish geographic location, we use the users’ current geo-IP location; this means, for example, that the users in the it-2007 graph are users who are today in Italy and were on Facebook on January 1, 2007 (most probably, American college students then living in Italy).

6We remark that the commercial value of such hardware is of the order of a few thousand dollars.
Possible questions in SNA

- Prestige
- Centrality
- Co-citation/co-occurrence
- Radius/diameter
- ...

Jaak Vilo and other authors
Prestige

• $p = \text{vector of prestige values}$
• $E$ - citations
• $E[i,j]$ – document $i$ cites document $j$

• Calculate a new prestige where prestige of citing documents is added to current prestige
“...we are involved in an 'infinite regress': [an actor's status] is a function of the status of those who choose him; and their [status] is a function of those who choose them, and so ad infinitum”

[Seeley, 1949]
Conclusions on course:

- Data Mining is a rich and diverse field
- Driven by data, curiosity, business value
- Algorithmics, Statistics, Visualisation, ...
- Decision support/actionable information