NDBI040: **Big Data Management and NoSQL Databases**
http://www.ksi.mff.cuni.cz/˜svoboda/courses/2016-1-NDBI040/

Lecture 12
Advanced Aspects

Lecturer: **Martin Svoboda**, Author: **Irena Holubová**
svoboda@ksi.mff.cuni.cz

3. 1. 2017

Charles University in Prague, Faculty of Mathematics and Physics
Czech Technical University in Prague, Faculty of Electrical Engineering

Managing Transactions

- Critics of NoSQL databases focus on the lack of support for transactions
- Business transaction
  - □ e.g., browsing a product catalogue, choosing a bottle of Talisker at a good price, filling in credit card information, and confirming the order
- System transaction
  - □ At the end of the interaction with the user
  - □ Locks are only held for a short period of time
- Business transaction = a series of system transactions

Managing Transactions

- **Offline concurrency** involves manipulating data for a business transaction that spans multiple data requests
  - □ Having a system transaction open for the whole business transaction is not usually possible
    - Long system transactions are not supported
- **Problems:**
  - □ Overwriting uncommitted data
    - More transactions select the same row and then update the row based on the value originally selected unaware of the other
  - □ Reading uncommitted data
    - A transaction accesses the same row several times and reads different data each time
- i.e., calculations and decisions may be made based on data that is changed
  - □ e.g., price list may be updated, someone may update the customer's address, changing the shipping charges, …

$$S = \begin{bmatrix} T1 & T2 \\ W(A) & \\ & W(B) \\ W(B) & \\ Com. & \\ & W(A) \\ & Com. \end{bmatrix}$$

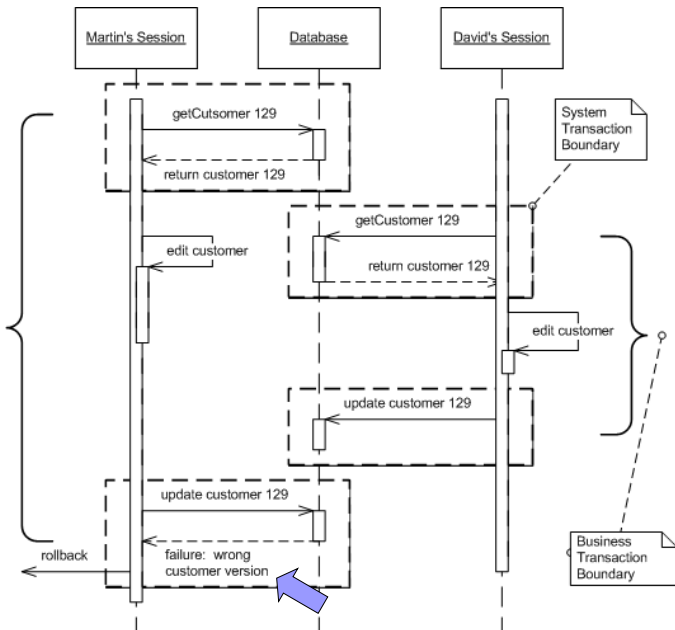overwriting uncommitted data
(blind write)

$$S = \begin{bmatrix} T1 & T2 \\ R(A) & \\ W(A) & \\ & R(A) \\ & W(A) \\ & R(B) \\ & W(B) \\ & Com. \\ R(B) & \\ W(B) & \\ Com. & \end{bmatrix}$$

reading uncommitted data
(dirty read)

Managing Transactions
## Optimistic Offline Lock

- Assumes that the <u>chance of conflict is low</u>
- A form of conditional update
  - Ensures that changes about to be committed by one session do not conflict with the changes of another session
- Pre-commit validation
  1. Client operation re-reads any information that the business transaction relies on
  2. It checks that it has not changed since it was originally read and displayed to the user
- Obtaining a lock indicating that it is okay to go ahead with the changes to the record data

Martin's Session | Database | David's Session

getCutsomer 129
return customer 129

System Transaction Boundary

getCustomer 129
return customer 129

edit customer

edit customer

update customer 129

update customer 129

rollback

failure: wrong customer version

Business Transaction Boundary

Managing Transactions
Pessimistic Offline Lock

- Problems of optimistic approach:
  - There might be many conflicts
  - The conflict can be detected at the end of a lengthy business transaction
- Pessimistic solution: allows <u>only one business transaction</u> at a time to <u>access data</u>
- Forces a business transaction to acquire a lock on each piece of data before it starts to use it
  - Once a business transaction begins, it surely completes
- Lock manager
  - Simple, single (for all business transactions), centralized (or based on the database in the distributed system)
- Standard issue: deadlock
  - Timeout for an application
    - Automatically rolled-back after a period of time of non responding
  - Timestamp attribute for a lock
   - Automatically released after a period of time
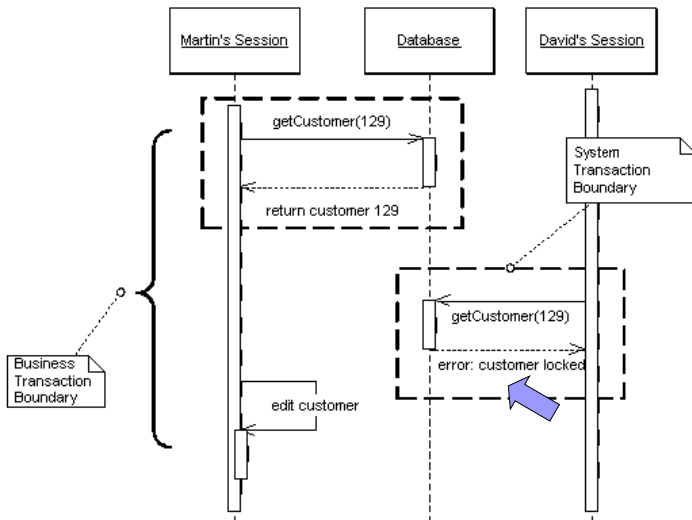
Managing Transactions
Coarse-grained Lock

- When objects are <u>edited as a group</u>
  - □ Logically related objects
  - □ e.g., a customer and its set of addresses
    - We want to lock any one of them
- A separate lock for individual objects presents a number of challenges
  - □ We need to find them all in order to lock them
    - Gets tricky as we get more locking groups
    - When the groups get complicated
      - □ Nested groups
- Idea: a single lock that covers many objects
  - □ A sophisticated lock manager
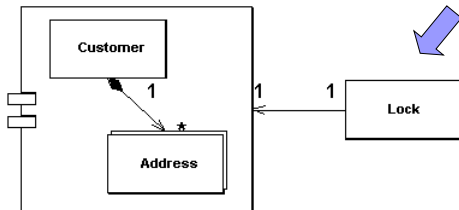
Managing Transactions
Implicit Lock

- Problem: forgetting to write a single line of code that acquires a lock $\Rightarrow$ entire offline locking scheme is useless
  - Failing to retrieve a read lock $\Rightarrow$ other transactions use write locks $\Rightarrow$ not getting up-to-date session data
  - Failing to use a version count $\Rightarrow$ unknowingly writing over someone's changes
  - Not releasing locks $\Rightarrow$ bring productivity to a halt
- Fact: If an item might be locked anywhere it must be locked everywhere
- Idea: <u>locks are automatically acquired</u>
  - Not explicitly by developers but implicitly by the application
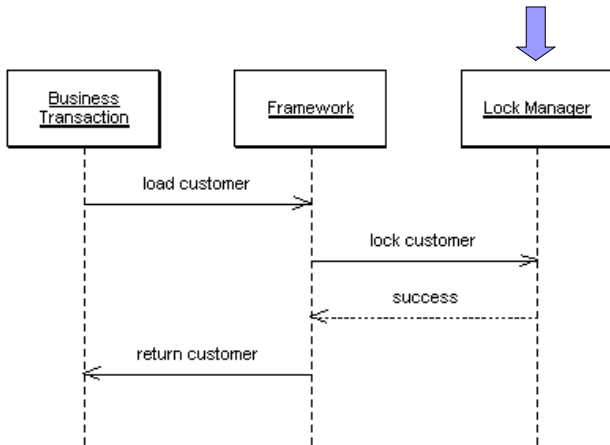
Performance Tuning
Goals

- MapReduce creates a bottleneck-free way of scaling out
- To reduce latency
    - Latency:
        - Non-parallel systems: time taken to execute the entire program
        - Parallel systems: time taken to execute the smallest atomic sub-task
    - Strategies:
        - Reducing the <u>execution time</u> of a program
        - Choosing the most <u>optimal algorithms</u> for producing the output
        - <u>Parallelizing</u> the execution of sub-tasks
- To increase throughput
    - Throughput = the amount of input that can be manipulated to generate output within a process
    - Non-parallel systems:
        - Constrained by the available resources (amount of RAM, number of CPUs)
    - Parallel systems:
        - "No" constraints
        - Parallelization allows for any amount of commodity hardware

Performance Tuning
Linear Scalability

- Typical horizontally scaled MapReduce-based model: linear scalability
    - "One node of a cluster can process x MBs of data every second $\rightarrow$ n nodes can process $x \times n$ amounts of data every second."
        - Time taken to process y amounts of data on a single node = t seconds
        - Time taken to process y amounts of data on n nodes = t / n seconds
- Assumption: tasks can be parallelized into equally balanced units

Performance Tuning

$$S(N) = \frac{1}{(1 - P) + \frac{P}{N}}.$$

Amdahl's Law

- Formula for <u>finding the maximum improvement</u> in performance of a system when a part is improved
  - P = the proportion of the program that is parallelized
  - $1 - P$ = the proportion of the program that cannot be parallelized
  - N = the times the parallelized part performs as compared to the non-parallelized one
    - i.e., how many times faster it is
      - e.g., the number of processors
    - Tends to infinity in the limit
- Example: a process that runs for 5 hours (300 minutes); all but a small part of the program that takes 25 minutes to run can be parallelized
  - Percentage of the overall program that can be parallelized: 91.6%
  - Percentage that cannot be parallelized: 8.4%
  - Maximum increase in speed: 1 / (1 – 0.916) = ~11.9 times faster
    - N tends to infinity

Performance Tuning

$L = kW$

Little's Law

- Origins in economics and queuing theory (mathematics)
- Analyzing <u>the load on stable systems</u>
  - □ Customer joins the queue and is served (in a finite time)
- "The average number of customers (L) in a stable system is the product of the average arrival rate (k) and the time each customer spends in the system (W)."
  - □ Intuitive but remarkable result
  - □ i.e., the relationship is not influenced by the arrival process distribution, the service distribution, the service order, or practically anything else
- Example: a gas station with cash-only payments over a <u>single</u> counter
  - □ 4 customers arrive every hour
  - □ Each customer spends about 15 minutes (0.25 hours) at the gas station
  - ⇒ There should be <u>on average 1 customer</u> at any point in time
  - ⇒ If more than 4 customers arrive at the same station, it would lead to a bottleneck

Performance Tuning
Message Cost Model

initialization

$$C = a + bN$$

linear dependence on size

- Breaks down <u>the cost of sending a message</u> from one end to the other in terms of its fixed and variable costs
  - □ *C* = cost of sending the message from one end to the other
  - □ *a* = the upfront cost for sending the message
  - □ *b* = the cost per byte of the message
  - □ *N* = number of bytes of the message
- Example: gigabit Ethernet
  - □ *a* is about 300 microseconds = 0.3 milliseconds
  - □ *b* is 1 second per 125 MB
    - Implies a transmission rate of 125 MBps.
  - □ 100 messages of 10 KB => take 100 × (0.3 + 10/125) ms = 38 ms
  - □ 10 messages of 100 KB => take 10 × (0.3 + 100/125) ms = 11 ms
  - □ A way to optimize message cost is to send as big packet as possible each time

0,08

0,8

Motivation for (Big) Data Visualization

- **Data visualization** = creation and studying of visual representation of data
  - Information abstracted in some schematic form
  - Including attributes, variables, …
- Purpose:
  - To communicate information clearly and effectively through graphical means
  - To help find the information needed more effectively and intuitively
- Both aesthetic form and functionality are required
- Even when data volumes are large, the patterns can be spotted quite easily (with the right data processing and visualization)
  - Simplification of Big Data management
 - Picking up things with the naked eye that would otherwise be hidden

Motivation

Similar motivation as for statistics but visualization can reveal/distinguish data/trends/patters, … which

| I		II		III		IV	
X	Y	X	Y	X	Y	X	Y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

Source: Tufte, Edward R (1983), *The Visual Display of Quantitative Information,* Graphics Press
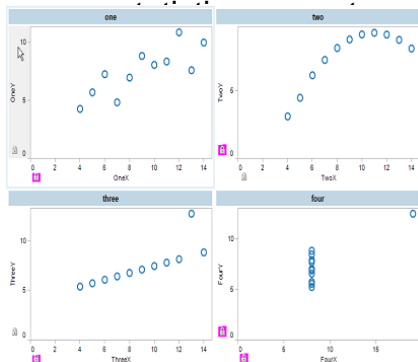
Four data sets with nearly identical linear model (mean, variance, linear regression line, …)

Motivation

	A	B
1	Year	Sales
2	1981	1.4622
3	1982	1.47004
4	1983	1.49253
5	1984	1.49118
6	1985	1.49722
7	1986	1.50138
8	1987	1.50008
9	1988	1.51493
10	1989	1.50781
11	1990	1.50899
12	1991	1.53037
13	1992	1.58137
14	1993	1.54299
15	1994	1.53307
16	1995	1.55845
17	1996	1.56213
18	1997	1.54488
19	1998	1.56927
20	1999	1.55305
21	2000	1.5571
22	2001	1.56235
23	2002	1.58847
24	2003	1.59309
25	2004	1.58303
26	2005	1.5947
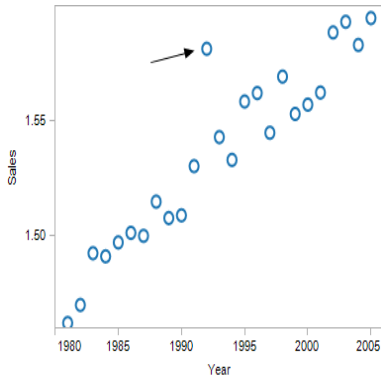
Find an outlier….

Data Visualization

- Information visualization has two equally important aspects
  - Structural modeling
    - Detection, extraction and simplification of the underlying information
  - Graphical representation
    - Transform initial representation into a graphical one which provides visualization of the structure
      - Different types of structures require different type of visualization
      - e.g., time series vs. hierarchical information

Big Data Visualization

- Decision about what technique to use became more difficult with Big Data
  - ☐ Visualization is needed to decide which portion of data to explore further
  - ☐ Visualization algorithms (i.e., graph drawing) should scale well to billions of entities (nodes)
    - The first application was probably the visualization of web-related data
      - ☐ i.e., pages, relations, traffic, …
  - ☐ New techniques may be needed
  - ☐ Trends might not be clear
  - ☐ Noise reduction might be even more necessary

Visualization Types
Data Relationships

- Scatter plot
  - Classical statistical diagram that lets us visualize relationships between numeric variables
  - Can carry additional information
- Matrix chart
  - Summarizes a multidimensional data set in a grid
- Network diagram
  - A set of objects (vertices) connected by edges
  - Visualization of the network is optimized to keep strongly related items in close proximity to each other

Visualization Types
Data Relationships

- **Correlation matrix (heat map)**
  - Combines data to quickly identify which variables are related
  - Shows how strong the relationship is between the variables

NBA per game performance of top 50 scorers

2008-2009 season

Source: databasebasketball
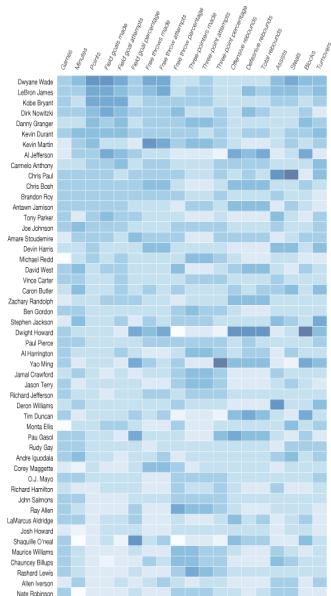
Visualization Types
Data Relationships

- Heat map is often combined with a dendrogram
 - Aggregates rows or columns based on their overall similarity into a tree structure
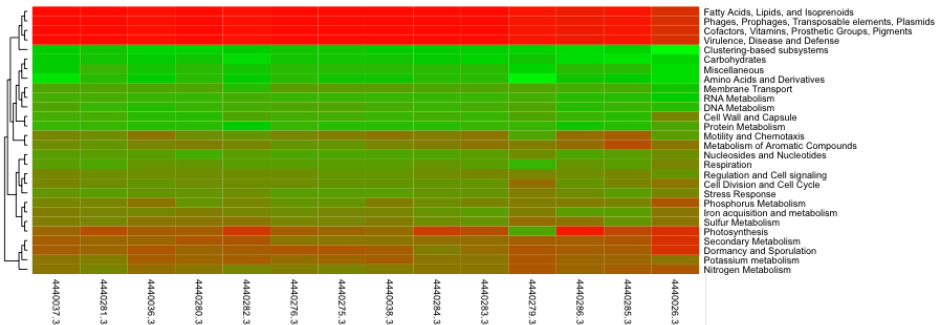
Visualization Types
Comparison of a Set of Values

- **Bar Chart**
  - Classical method for numerical comparisons
  - Histograms
  - Box plot (box-and-whisker plots)
    - Five statistics (minimum, lower quartile, median, upper quartile and maximum) summarizing the distribution of a set of data
- **Bubble chart**
  - Circles in a bubble chart represent different data values
    - The area of a circle corresponding to the value
  - The positions of the bubbles do not mean anything
    - Designed to pack the circles together with relatively little wasted space
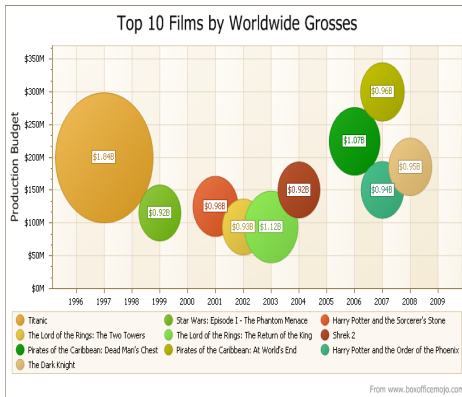
Experiment No.

Visualization Types
Trends over Time

- Line graph
  - Classical method for visualizing continuous change
- Stack graph
  - Visualizing change in a set of items
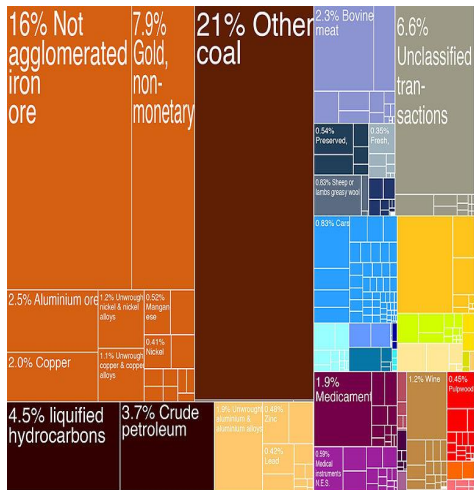 - The sum of the values is as important as the individual items
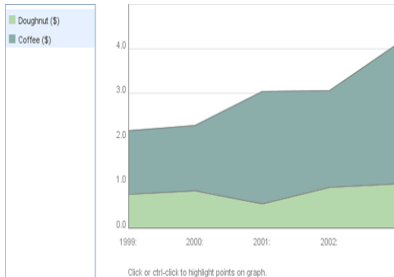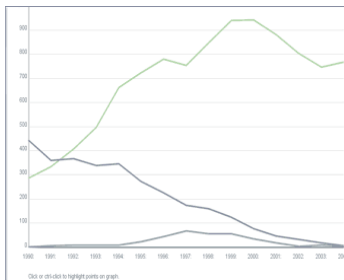
Visualization Types

Parts of a Whole

- Pie Chart
  - Percentages are encoded as "slices" of a pie, with the area corresponding to the percentage
- Treemap
  - Visualization of hierarchical structures
  - Effective in showing attributes of leaf nodes using size and color coding
  - Enable to compare nodes and sub-trees at varying depth

Economy of Australia

Visualization Types
Text Analysis

- ## Tag cloud
  - Visualization of word frequencies
    - i.e., how frequently words appear in a given text

Which Visualization Technique to Use?

- New visualization software is capable of "guessing" the correct visualization based on the characteristics of the data
  - One-dimensional data $\Rightarrow$ bar chart
  - Two-dimensional data $\Rightarrow$ scatter plot
  - N-dimensional data $\Rightarrow$ multiple scatter plots, matrix chart, …
  - Data with coordinates $\Rightarrow$ map-based charts
- Offers options
- Trend: to simplify the process for common users

Big Data Visualization

- The goal of visualizing Big Data is usually to make sense of a large amount of interlinked information
- In interconnected data the connections between objects are difficult to organize on a linear layout
  - **Circular representations**
  - **Network diagrams**
- Typical "topologies" one can encounter (a bit confusing term based on Manuel Lima's "Visual Complexity" – see references) include arc diagrams, centralized burst, centralized ring, globe, circular ties or radial convergence
  - And many more…

Arc Diagram

- Vertices are placed on a line and edges are drawn as semicircles
- Arcs represent relationships
  - Colors can encode, e.g., distance
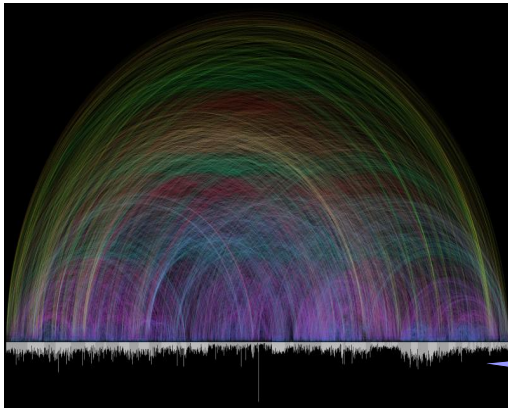
A map of 63,799 cross-references found in the Bible. The bottom bars represent number of verses in the given chapter. Color of arcs represents the distance between the two chapters.

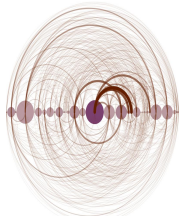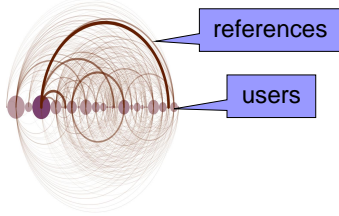http://www.chrisharrison.net/index.php/Visualizations/BibleViz

grey/white = book

Arc Diagram
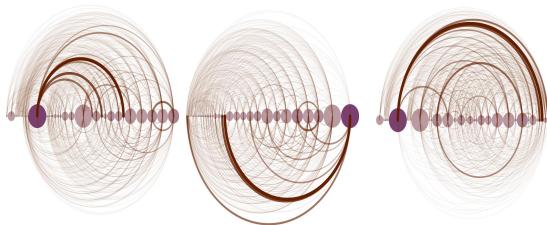
Sorted by the amount of incoming references

Sorted by the amount of outgoing references

Sorted by rate of incoming/outgoing references

Sorted by user name

Unsorted

references

users

- Visualization of IRC communication behavior: Who is talking to whom?
- Arcs are directional and drawn clockwise:
    - □ In the upper half of a graph they point from left to right, in the bottom half from right to left
    - □ Arc strength corresponds to the number of references from the source to the target
- This visualization favors strong social connections over sociability: Frequent references between the same two users feature more prominently than combined references from several sources to a single target.
- http://datavis.dekstop.de/irc_arcs/

Centralized Burst
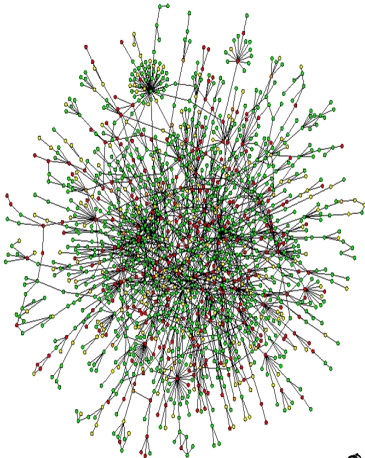
- Visualization with strong central tendency
- Can reveal highly connected objects (hubs) which usually correspond to objects with high importance
  - □ e.g., in a gene network, hubs are interesting points for targeting new drugs
    - Disabling a central gene probably will not allow the organism to adapt

A map of protein-to-protein interactions of a yeast
source: H. Jeong. et al. "Lethality and Centrality in Protein Networks", Nature, no. 411, 2011: 41-42

Centralized Ring

- Topology suitable for situations where we inspect a relation of multiple objects to one object
- Not very suitable for Big Data
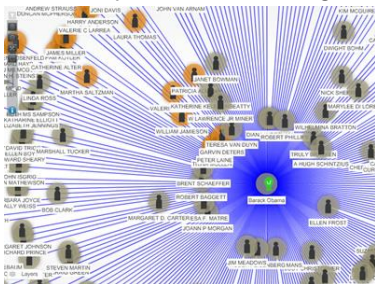
A sociogram of individual donations in Asheville, North Carolina, to Barack Obama's 2008 presidential campaign.
http://www.visualcomplexity.com/vc/project.cfm?id=613
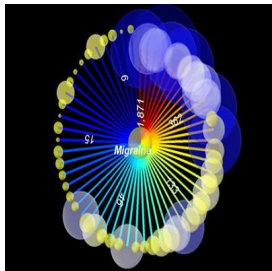
A map of genetic overlap between migraine and about 60 other diseases. Each of the circles represents a different disease, and the size of the circle corresponds to the size of patient samples, ranging from 46 to 136,000, of those suffering from the disease.
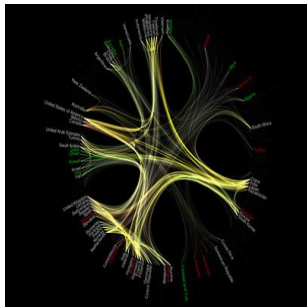
Globe

- Globe visualizations are basically projections of other topologies on a globe
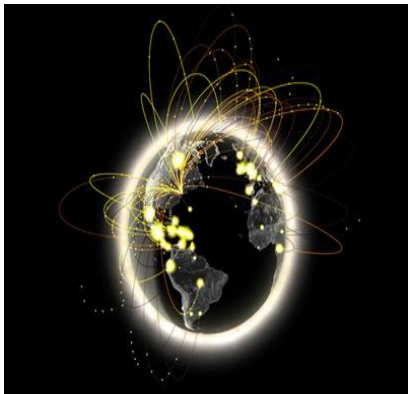
- The global exchange of information in real time by visualizing volumes of long distance telephone and IP data flowing between New York and cities around the world.
- How does the city of New York connect to other cities? With which cities does New York have the strongest ties and how do these relationships shift with time? How does the rest of the world reach into the neighborhoods of New York? The size of the glow on a particular city location corresponds to the amount of IP traffic flowing between that place and New York City. A greater glow implies a greater IP flow.
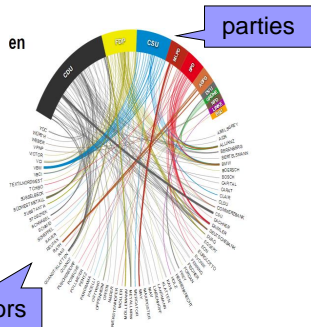
http://www.aaronkoblin.com/work/NYTE/index.html

Radial Convergence

- Also known as radial chart
- Actually a 360 arc diagram

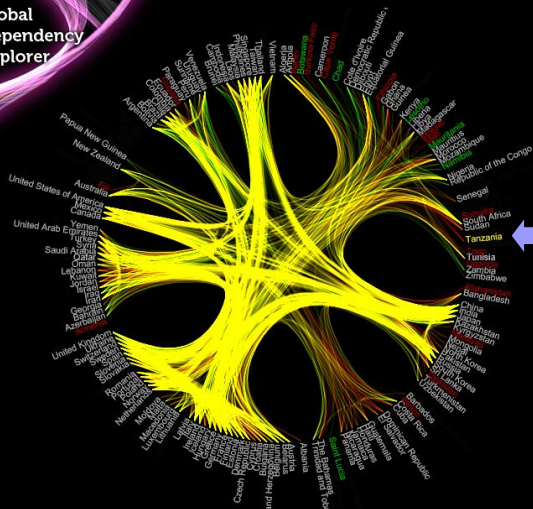Tracking the commercial ties between most countries across the globe.
http://cephea.de/gde/

parties

donators

Money flow from private donators to parties in the German Bundestag (house of the parliament).
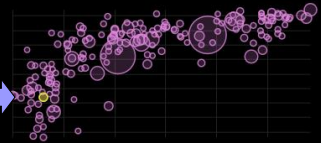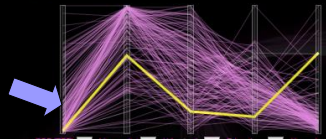http://labs.vis4.net/parteispenden/

global
dependency
explorer

search country:

bubblechart

x: Median age   y: Life expects   size: Population

parallel coordinates

Polyglot Persistence

- Different databases are designed to solve different kinds of problems
- Using a single database engine for all of the requirements usually leads to partially non-performant solutions
- Example: e-commerce
  - Many types of data
    - Business transactions, session management data, reporting, data warehousing, logging information, …
 - Do not need the same properties of availability, consistency, or backup requirements
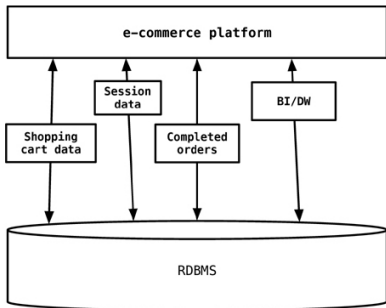
Polyglot Persistence

- Polyglot programming (2006)
    - Applications should be written in a mix of languages
    - Different languages are suitable for tackling different problems
- Polyglot persistence
    - Hybrid approach to persistence
    - e.g., a data store for the shopping cart which is highly available vs. finding products bought by the customers' friends
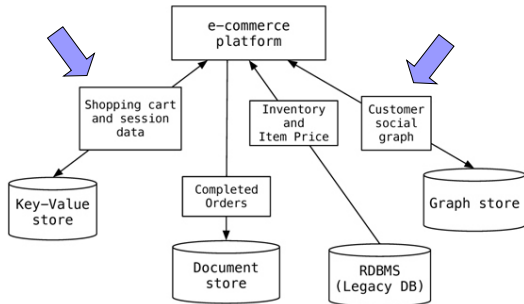
Polyglot Persistence

- There may be other applications in the enterprise
  - e.g., the graph data store can serve data to applications that need to understand which products are being bought by a certain segment of the customer base
⇒ Instead of each application talking independently to the graph database, we can wrap the graph database into a service
  - Assumption:
    - Nodes can be saved in one place
    - Queried by all the applications
  - Allows for the databases inside the services <u>to evolve without having to change the dependent applications</u>