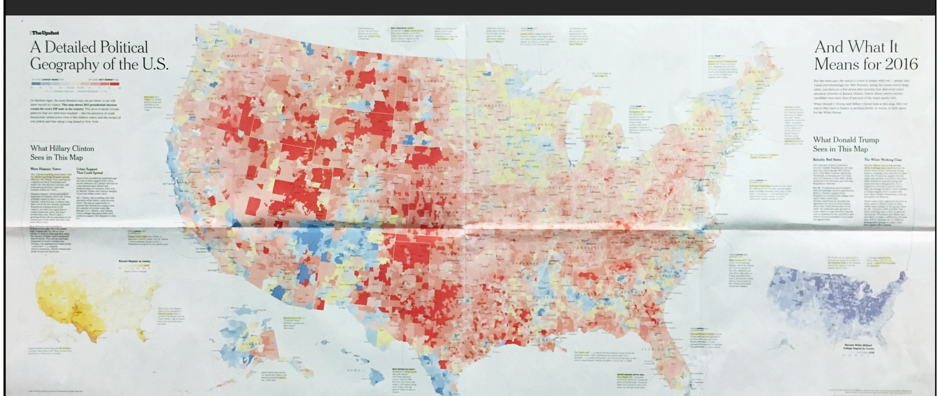
Deconstructing Visualizations

Maneesh Agrawala

CS 448B: Visualization
Fall 2017

NYT Election 2016 (based on 2012)



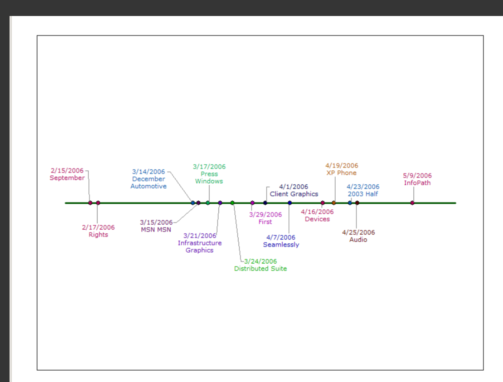
Last Time: Spatial Layout

Problem

Input: Set of graphic elements (scene description)

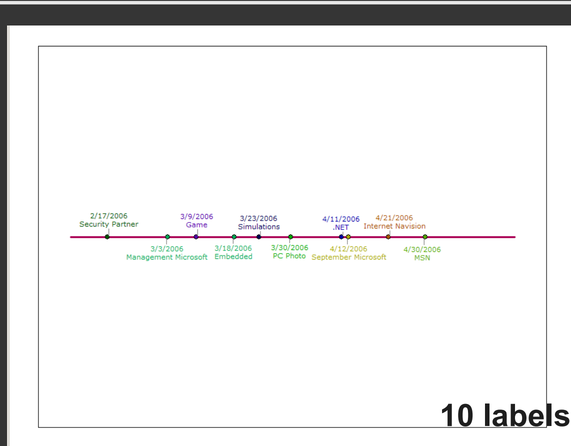
Goal: Select visual attributes for elements

- **Position**
- Orientation
- Size
- Color
- …



Direct Rule-Based Methods

Rule-based timeline labeling



- Alternate above/below line
- Center labels with respect to point on line

Pros and cons

Pros

- Designed to run extremely quickly
- Simple layout algorithms are easy to code

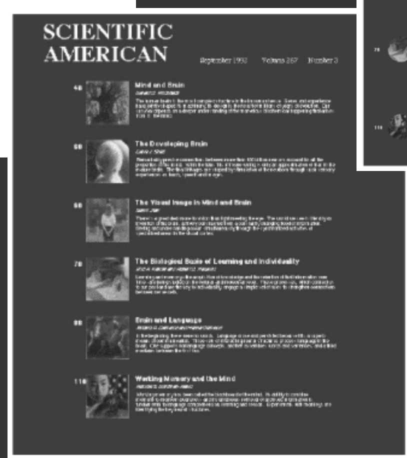
Cons

- Complex layouts require large rule bases with lots of special cases

Linear Constraint Satisfaction

Page layout example [Weitzman and Wittenburg 94]

```
(Defrule (Make-Article The-Grammar)
  Article -> Text Text Text Number Image
     0          1    2    3     4      5
  (Author-Of 2 1)
  (Description-Of 4 1)
  (Page-Of 4 1)
  (Image-Of 5 1)
  (article-name 0) = r
  (article-image 0) = 5
  :OUT
  (right-of 1 5)
  (top-aligned 1 5)
  (top-aligned 5 4)
  (spaced-below 2 1)
  (spaced-below 3 2)
)
```



Pros and cons

Pros

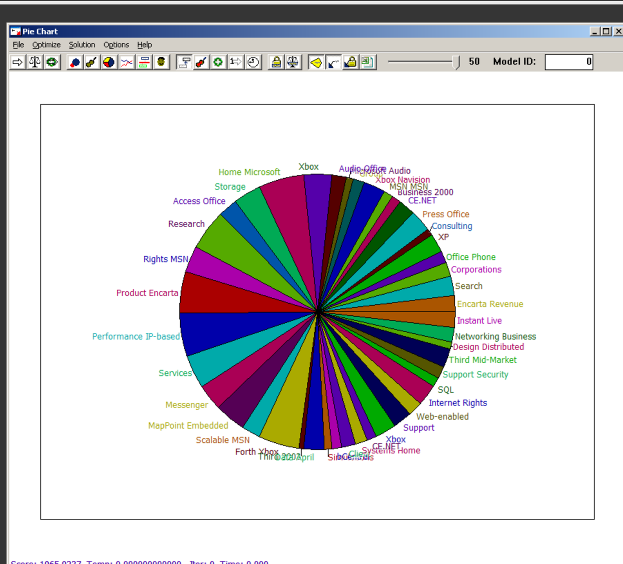
- Often run fast (at least one-way constraints)
- Constraint solving systems are available online
- Can be easier to specify relative layout constraints than to code direct layout algorithm

Cons

- Easy to over-constrain the problem
- Constraint solving systems can only solve some types of layout problems
- Difficult to encode desired layout in terms of mathematical constraints

Optimization

Demo



Layout as optimization

Scene description

- **Geometry:** polygons, bounding boxes, lines, points, etc.
- **Layout parameters:** position, orientation, scale, color, etc.

Large design space of possible layouts

To use optimization we will specify …

- **Initialize/Perturb functions:** Form a layout
- **Penalty function:** Evaluate quality of layout
- **.. and find layout that minimizes penalty**

Optimization algorithms

There are lots of them:

line search, Newton's method, A*, tabu, gradient descent, conjugate gradient, linear programming, quadratic programming, simulated annealing, …

Differences

- Speed
- Memory
- Properties of the solution
- Requirements

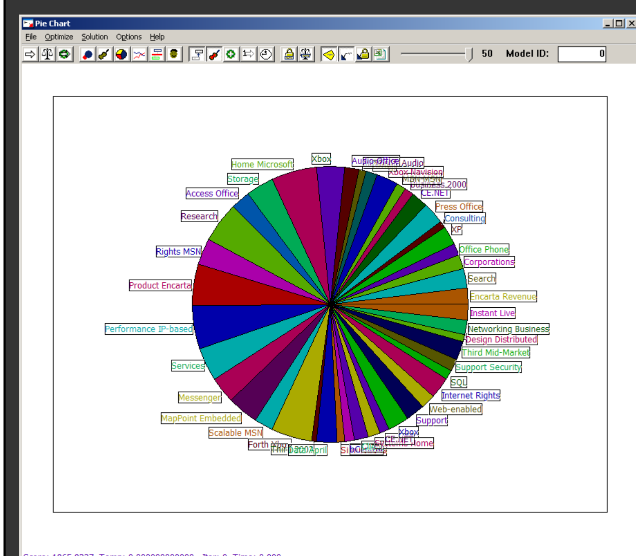
Simulated annealing

currL ← Initialize()	Form initial layout
while**(! termination condition)**	
newL ← Perturb(***currL***)	Perturb to form new layout
currE ← Penalty(***currL***)	Evaluate quality of layouts
newE ← Penalty(***newL***)	
if**((*newE* < *currE*) or**	Always accept lower penalty
(rand[0,1) < $e^{-\Delta E/T}$**))**	Small probability of accepting higher penalty
then ***currL* ← *newL***	
Decrease(*T*)	

Perturb: Efficiently cover layout design space

Penalty: Describes desirable/undesirable layout features

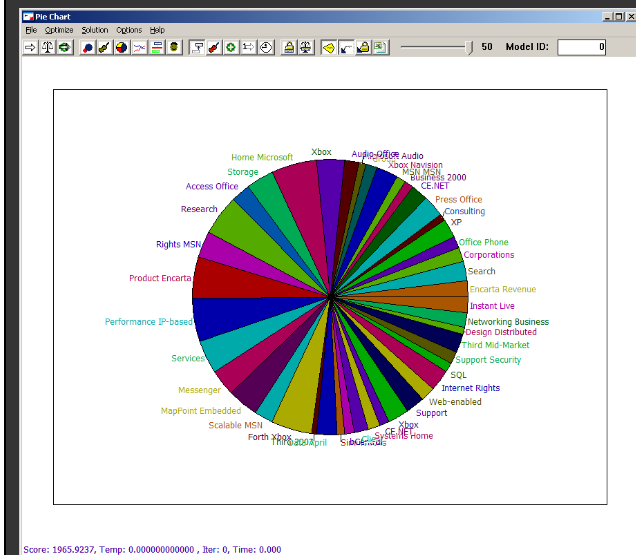
Scene description



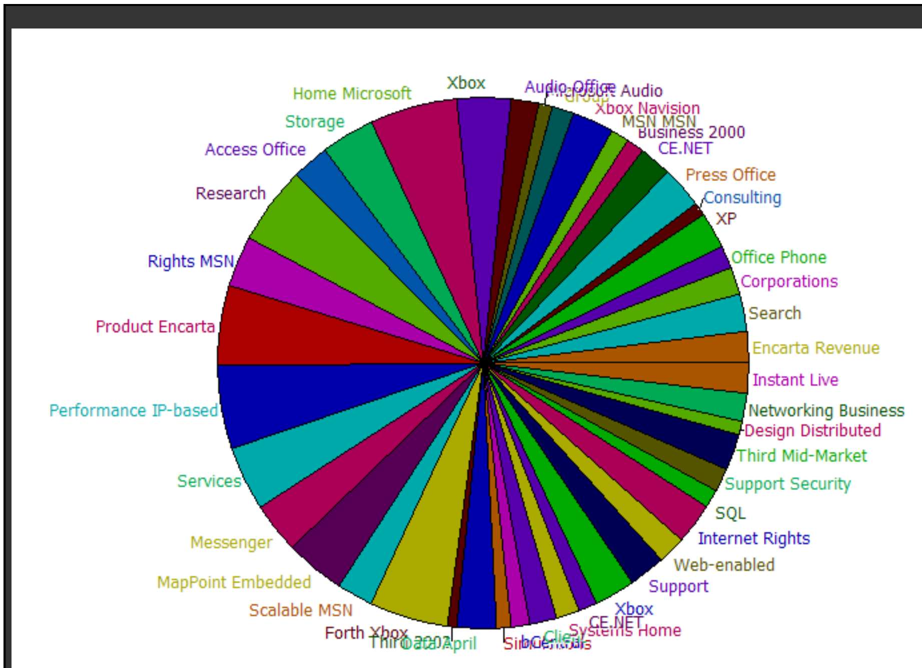
Geometry

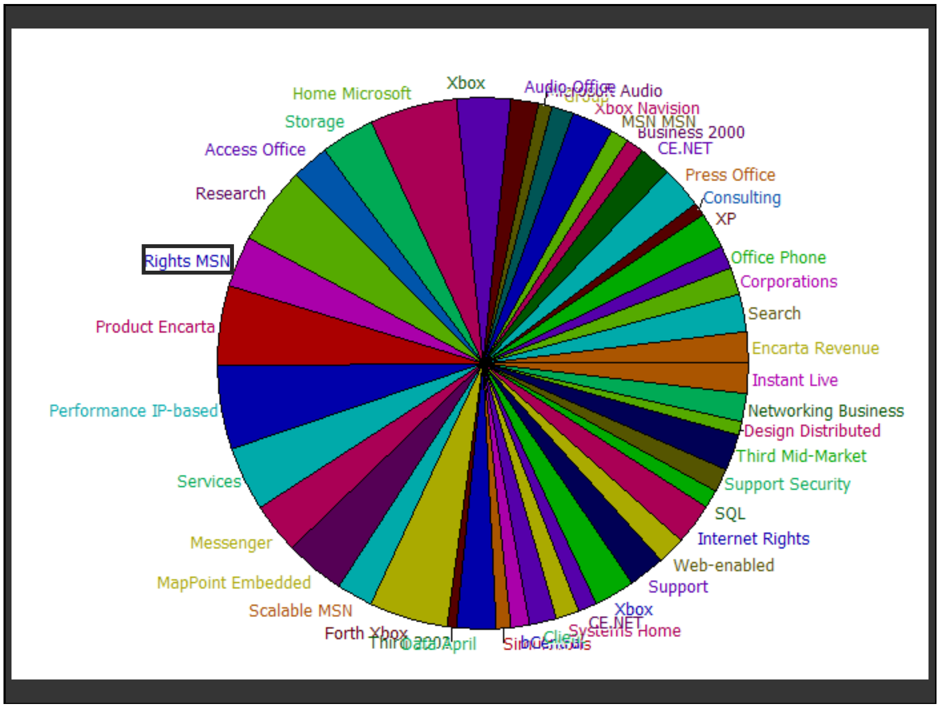
- **Pie slices**
  anchors for labels
- **Labels**
  bounding boxes

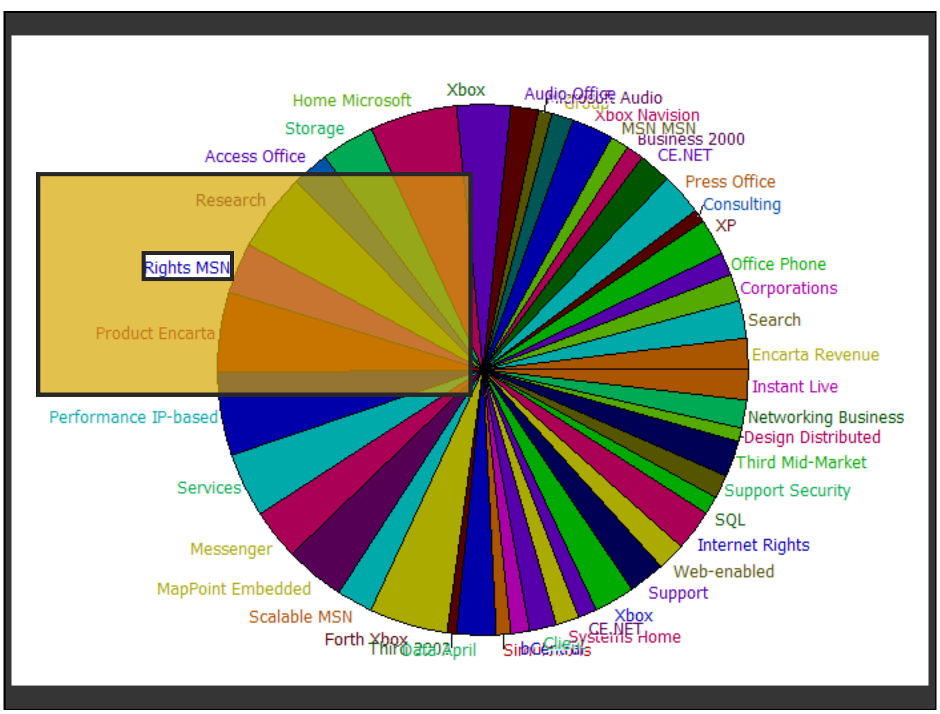
Layout parameters

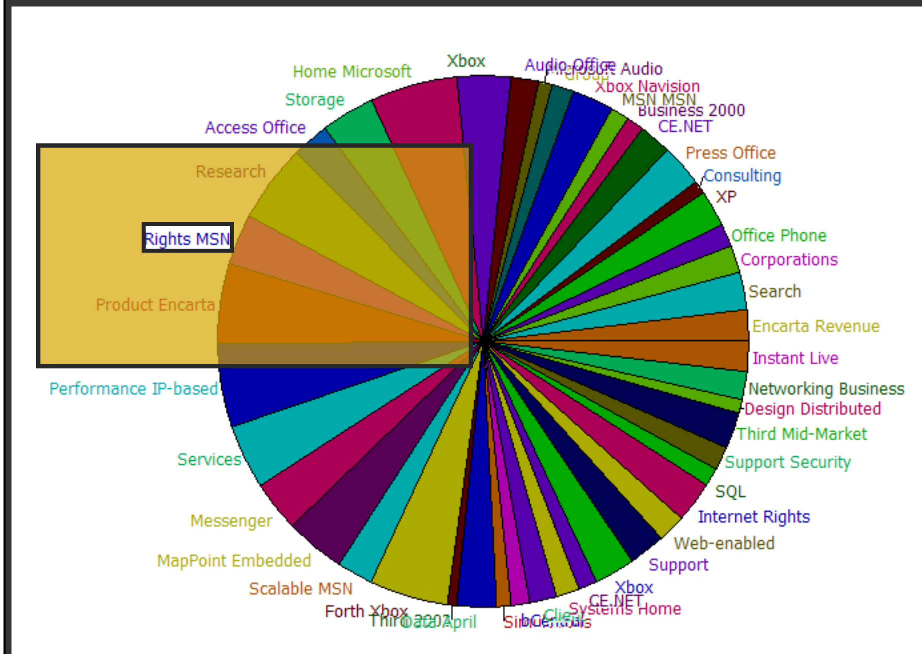


- **Position *(x, y)***
- **Leader line**
- **Word wrap**
- Color
- Alignment
- Orientation
- Scale

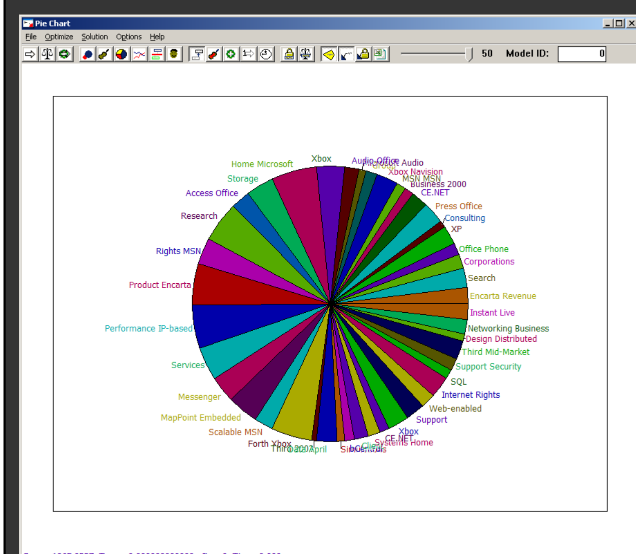



Home Microsoft
Xbox
Audio
Xbox Navision
MSN MSN
Business 2000
CE.NET
Storage
Access Office
Research
Press Office
Consulting
XP
Rights MSN
Office Phone
Corporations
Search
Product Encarta
Encarta Revenue
Instant Live
Performance IP-based
Networking Business
Design Distributed
Third Mid-Market
Services
Support Security
SQL
Messenger
Internet Rights
Web-enabled
MapPoint Embedded
Support
Scalable MSN
Xbox
CE.NET
Forth Xbox
Systems Home


Home Microsoft
Xbox
Audio
Xbox Navision
MSN MSN
Business 2000
CE.NET
Storage
Access Office
Research
Press Office
Consulting
XP
Rights MSN
Office Phone
Corporations
Search
Product Encarta
Encarta Revenue
Instant Live
Performance IP-based
Networking Business
Design Distributed
Third Mid-Market
Services
Support Security
SQL
Messenger
Internet Rights
Web-enabled
MapPoint Embedded
Support
Scalable MSN
Xbox
CE.NET
Forth Xbox
Systems Home



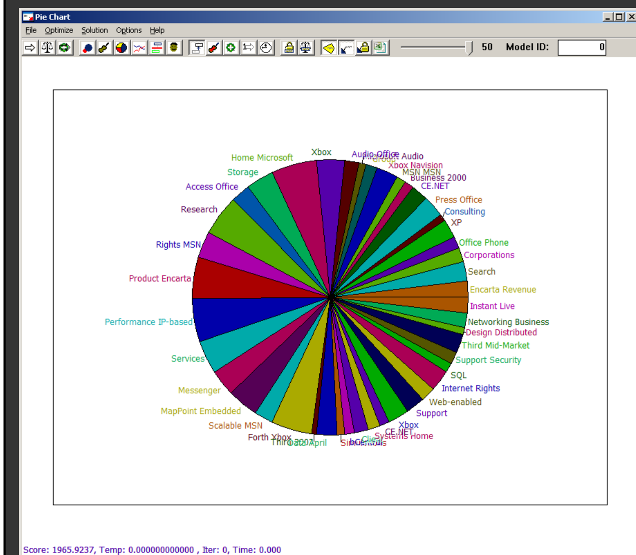
Many dimensions → large space



- **Position *(x, y)***
- Leader line
- Word wrap
- Color
- Alignment
- Orientation
- Scale

2D x 50 labels → 100D space

Penalties



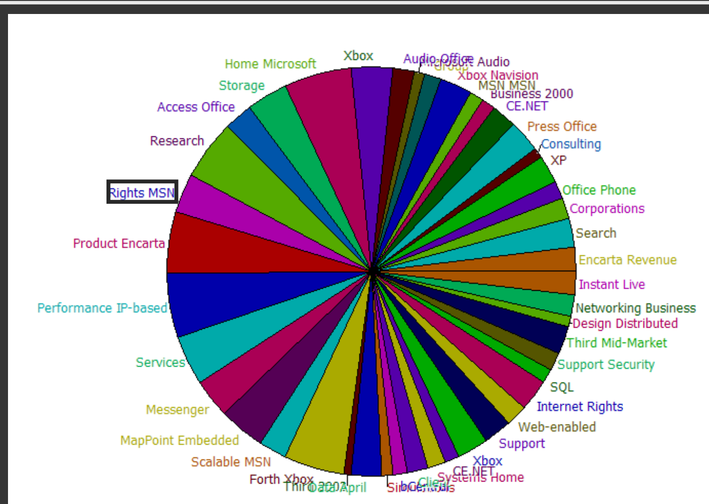
Overlap & Distance
- Label – anchor slice
- Label – other slices
- Label – label

Leader lines
- Length
- Intersections

Word Wrap

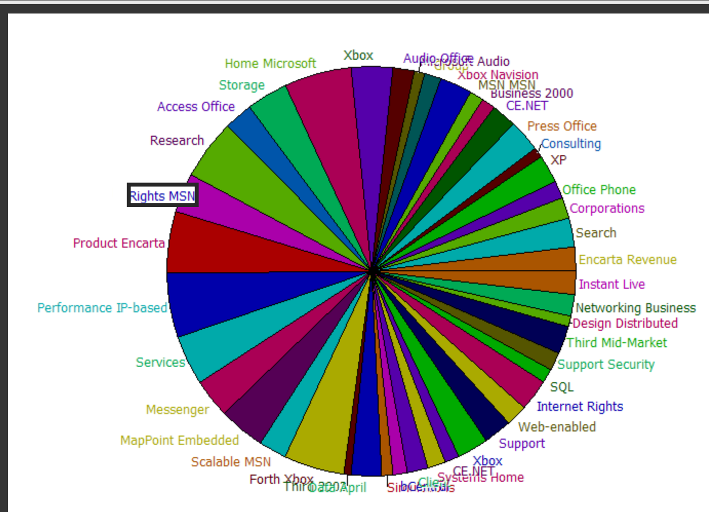
Annealing minimizes sum of all penalties

Overlap: Label – Anchor Slice



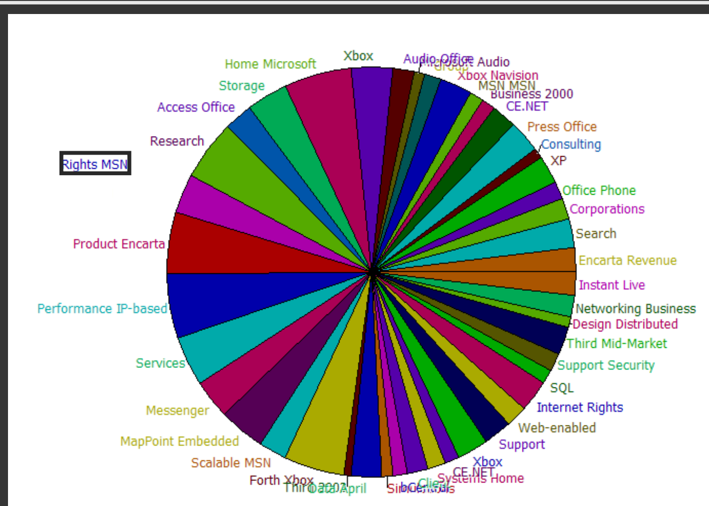
Avoid partial overlap: No penalty if fully inside /outside

Overlap: Label – Anchor Slice



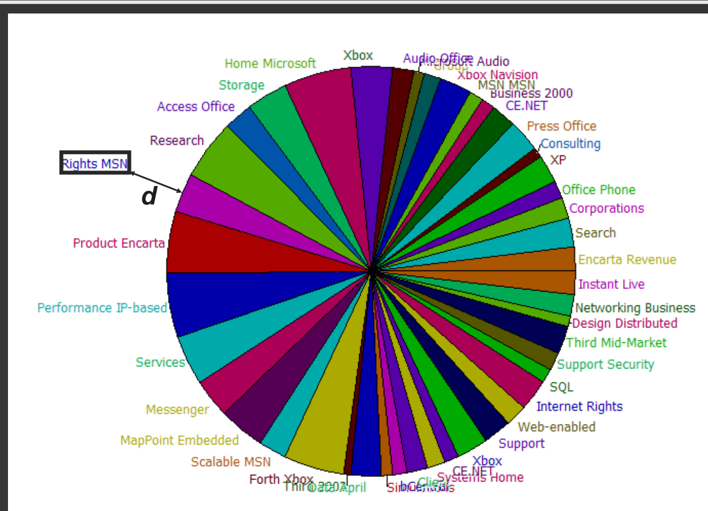
Penalize partial overlap by overlap amount

Distance: Label – Anchor Slice



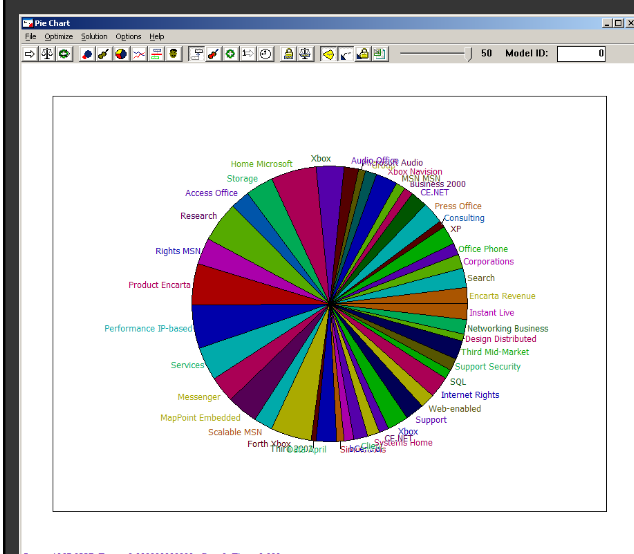
Ensure label near center of edge of anchor slice

Distance: Label – Anchor Slice



Minimize distance *d*

Penalties



Overlap & Distance

- Label – anchor slice
- Label – other slices
- Label – label

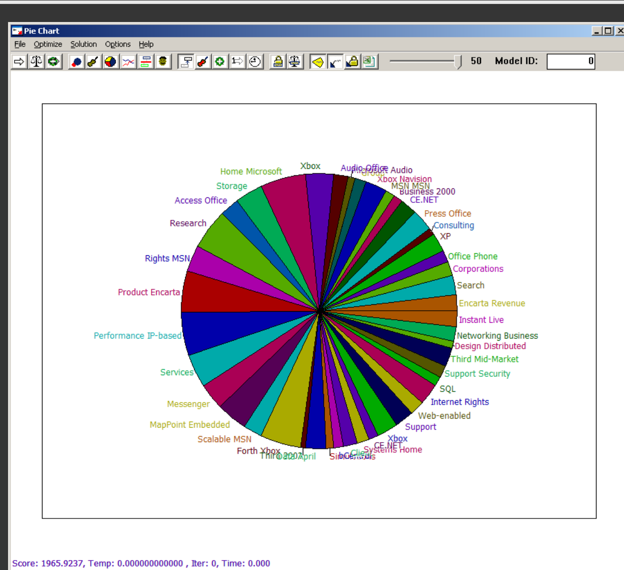
Leader lines

- Length
- Intersections

Word Wrap

Annealing minimizes sum of all penalties

Demo



Pros and cons

Pros

- Much more flexible than linear constraint solving systems

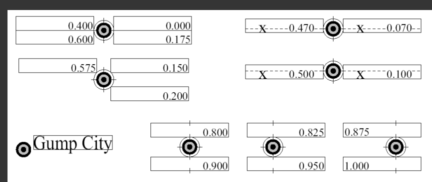
Cons

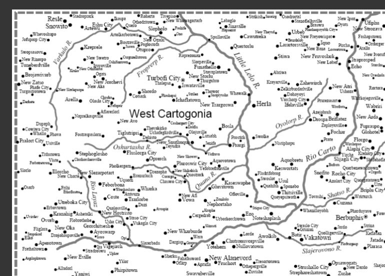
- Can be relatively slow to converge
- Need to set penalty function parameters (weights)
- Difficult to encode desired layout in terms of mathematical penalty functions

Design principles

Sometimes specified in design books

- Tufte, Few, photography manuals, cartography books …
- Often specified at a high level
- Challenge is to transform principles into constraints or penalties





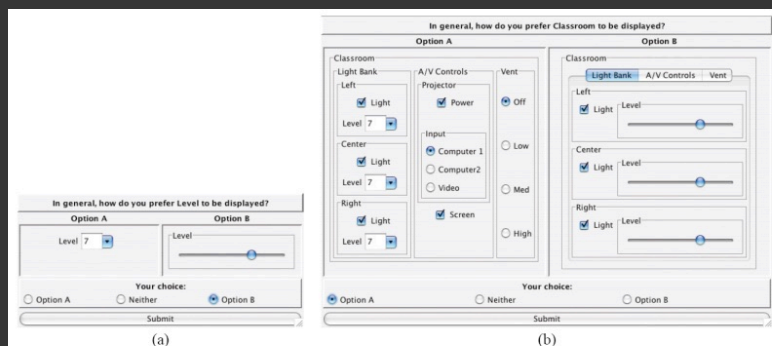
Cartographer Eduard Imhof's labeling heurists transformed into penalty functions for an optimization based point labeling system [Edmondson 97]

Example-Based Methods

Preference elicitation [Gajos and Weld 05]

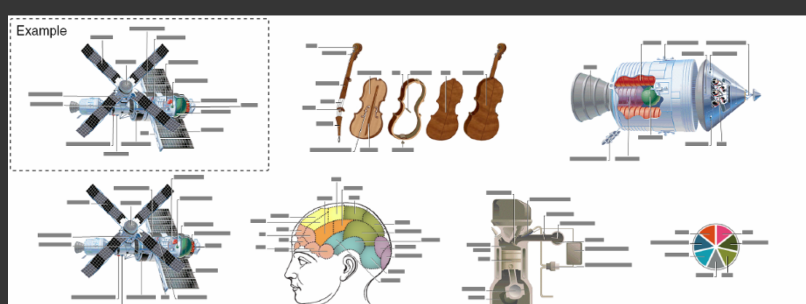
Learn characteristics of good designs

- Generate designs based on a parameterized design space
- Ask designers if they are good or bad
- Learn good parameters values based on responses



Nonlinear Inverse Opt. [Vollick et al. 07]

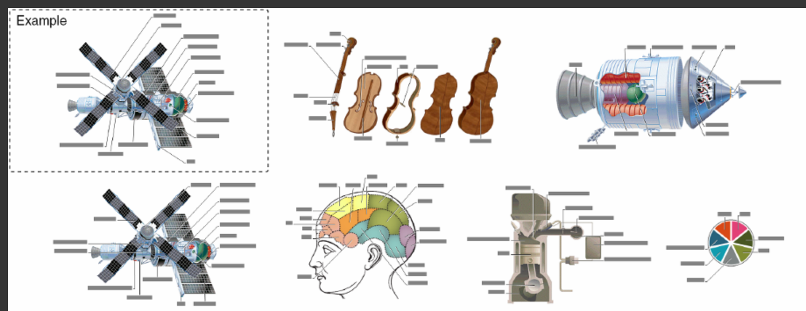
Learn label layout style from single example



Horizontal/Vertical

Nonlinear Inverse Opt. [Vollick et al. 07]

Learn label layout style from single example



Parallel Leader Lines

Pros and cons

Pros

- Often much easier to specify desired layout via examples

Cons

- Usually requires underlying model
- Model will constrain types of layouts possible
- Large design spaces likely to require lots of examples to learn parameters well

Announcements

Final project

Design new visualization method (e.g. software)

- Pose problem, Implement creative solution
- Design studies/evaluations less common but also possible (talk to us)

Deliverables

- Implementation of solution
- 6-8 page paper in format of conference paper submission
- Project progress presentations

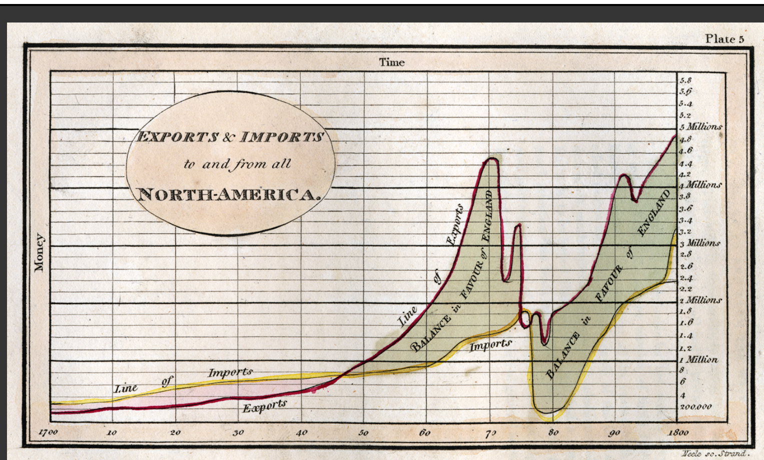
Schedule

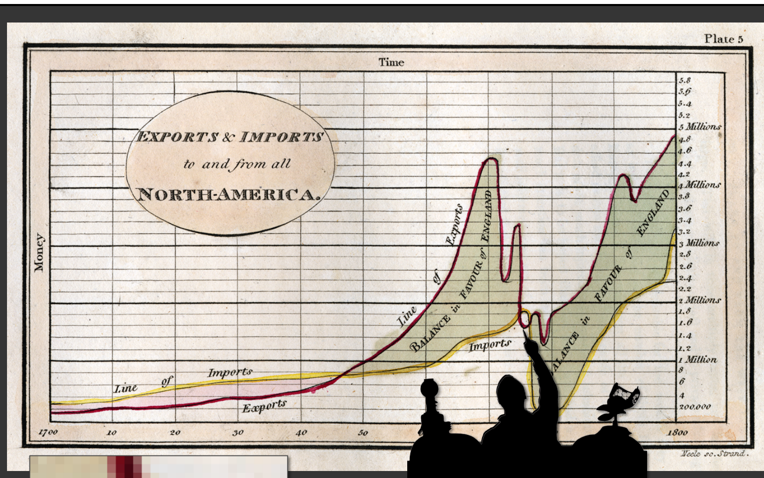
- Project proposal: Mon 11/6
- Project progress presentation: 11/13 and 11/15 in class (3-4 min)
- Final poster presentation: 12/6 Location: Lathrop 282
- Final paper: 12/10 11:59pm

Grading

- Groups of up to 3 people, graded individually
- Clearly report responsibilities of each member

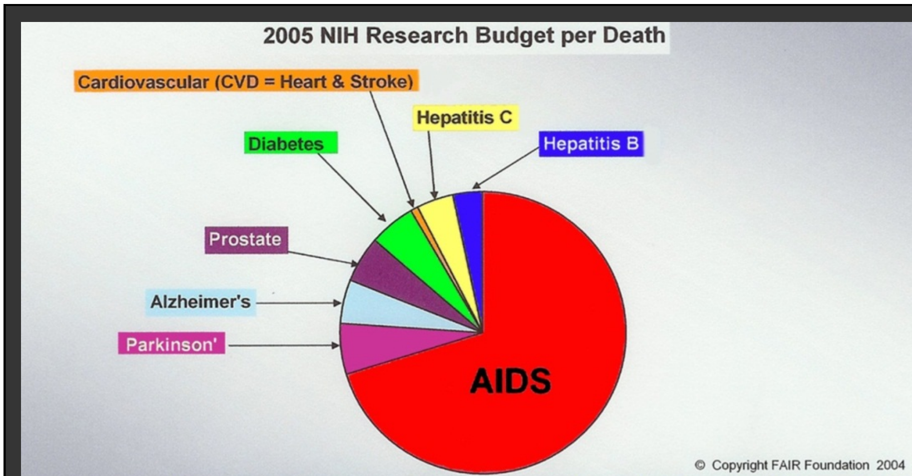
Deconstructing Visualizations

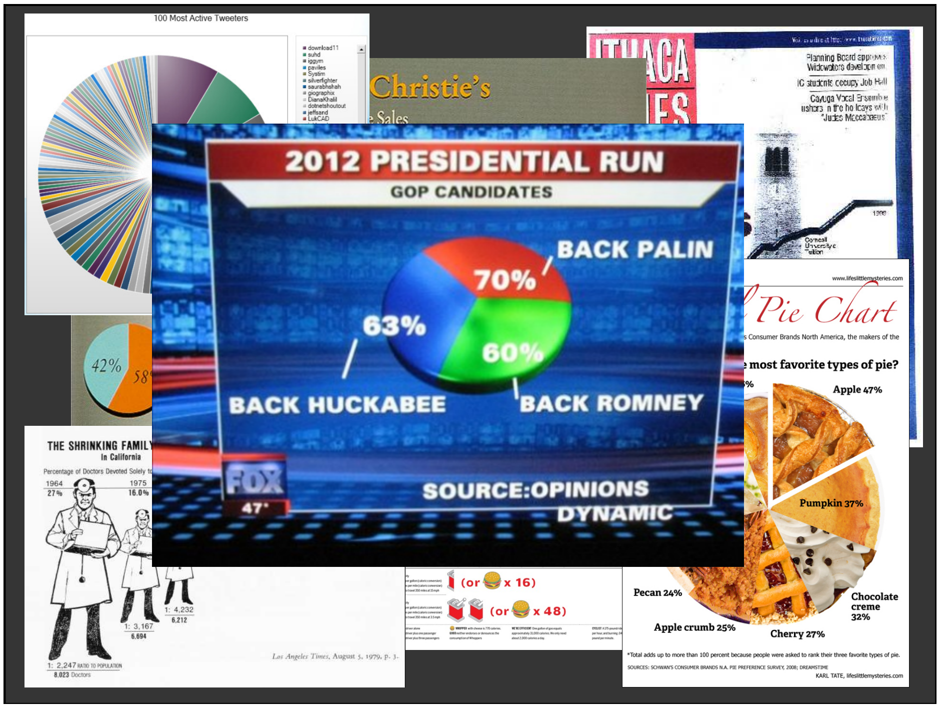


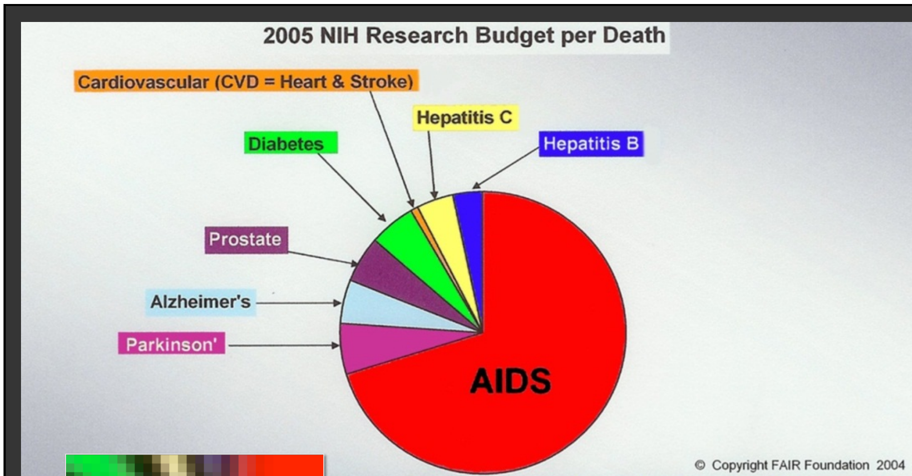


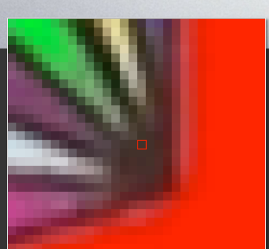
Pixels are poor representation

Hard for machines to retrieve data





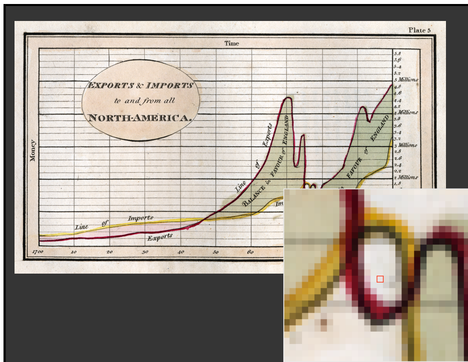


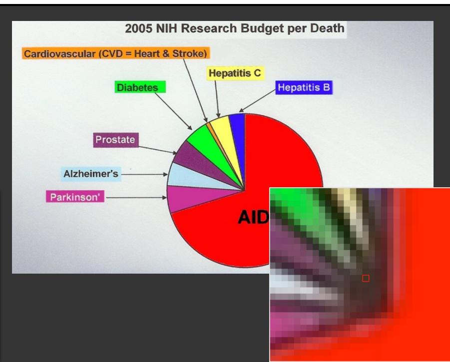


Pixels are poor representation

Hard for machines to retrieve data

Hard for people to manipulate

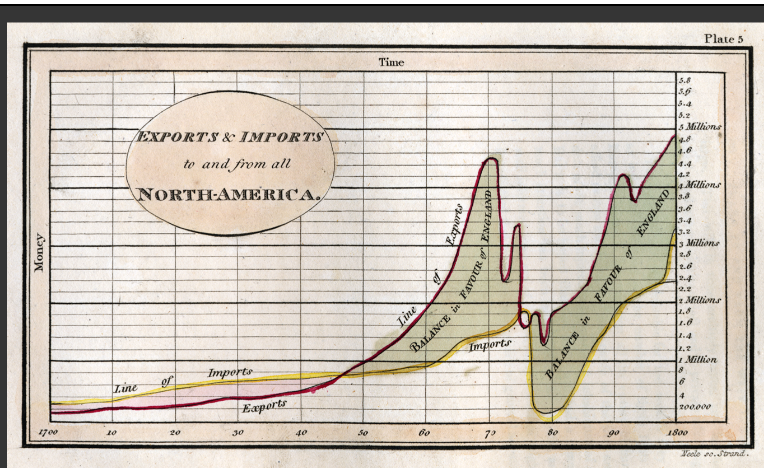




Pixels are a poor representation of charts and graphs
Cannot index, search, manipulate or interact with the data

Goal: ***Reconstruct higher-level representation*** *of charts and graphs that lets machines and people* ***redesign, reuse*** *and* ***revitalize*** *them*

What is a good representation?



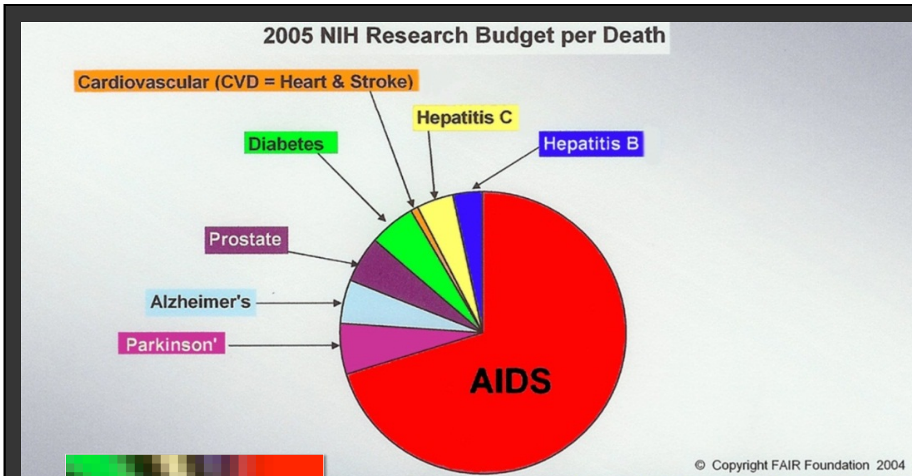
Data

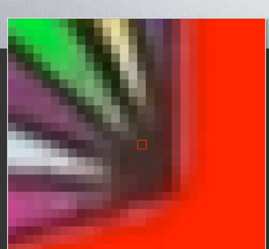
Year	Exports	Imports
1700	170,000	300,000
1701	171,000	302,000
1702	176,000	303,000
1703	180,000	312,000
1704	187,000	319,000
…	…	…

Marks
Mappings

mark: lines

Year → x-pos (Q)
Exports → y-pos (Q)
Imports → y-pos (Q)
Exports → color (N)
Imports → color (N)





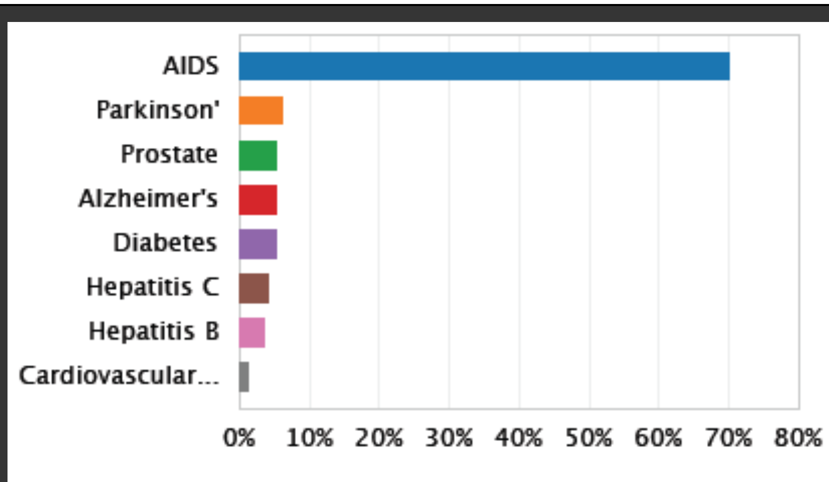
Data

Disease	Budget
Aids	70.0%
Alzheimer's	5.0%
Cardiovascular	1.1%
Diabetes	4.8%
Hepatitus B	4.1%
Hepatitus C	3.8%
Parkinson'	6.0%
Prostate	5.2%

Marks
Mappings

mark: areas

Budget → angle (Q)
Disease → color (N)



Data

Disease	Budget
Aids	70.0%
Alzheimer's	5.0%
Cardiovascular	1.1%
Diabetes	4.8%
Hepatitus B	4.1%
Hepatitus C	3.8%
Parkinson'	6.0%
Prostate	5.2%

Marks

mark: lines

Mappings

Budget → length (Q)
Disease → color (N)

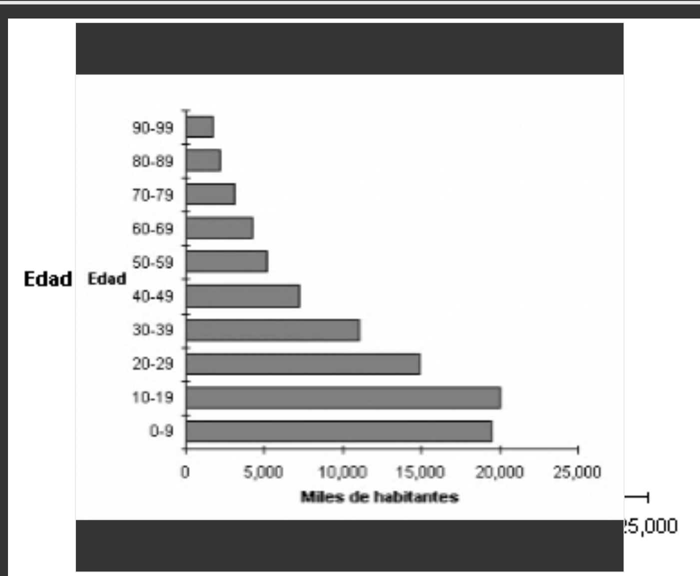
Approach

Classification: Determine chart type
Mark extraction: Retrieve graphical marks
Data extraction: Retrieve underlying data table

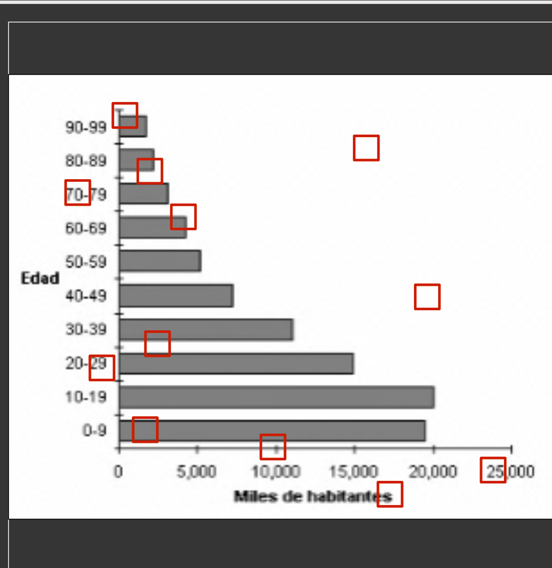
57

Classification

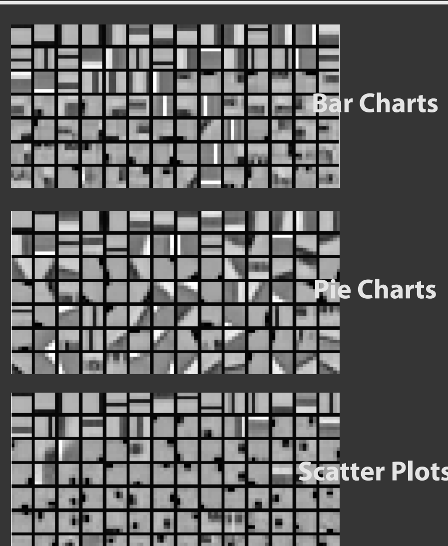
Training the Classifier



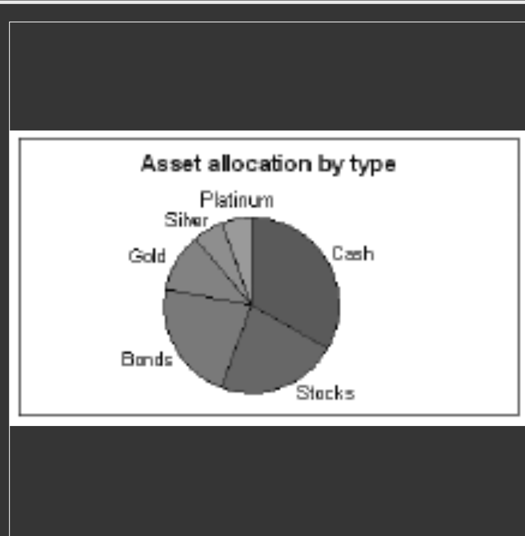
Training the Classifier



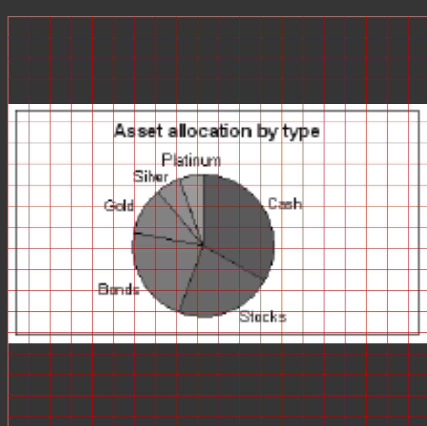
Training the Classifier



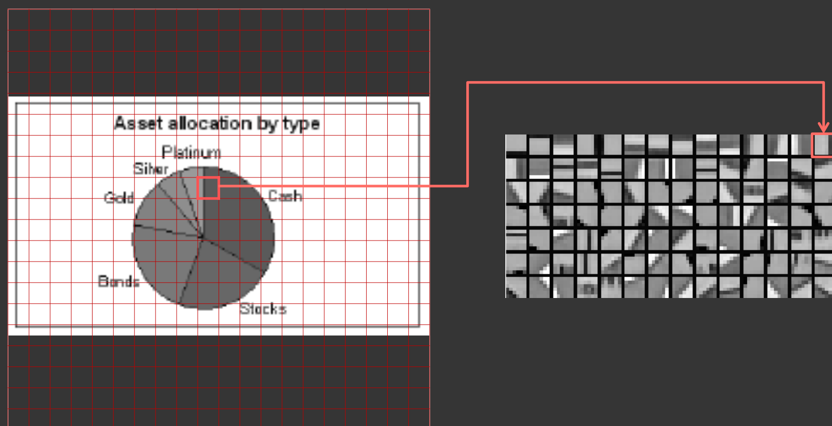
Classifying an Input Image



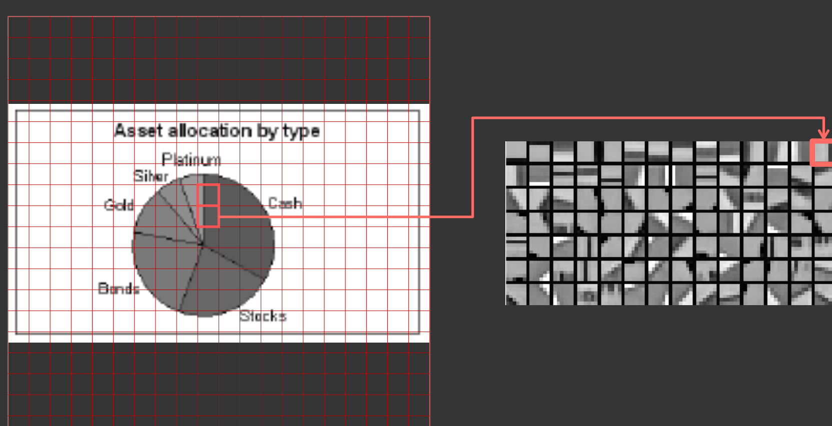
Classifying an Input Image



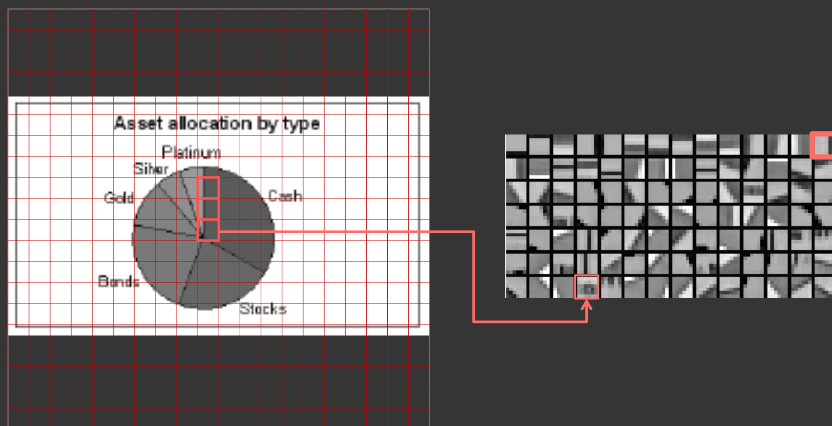
Classifying an Input Image



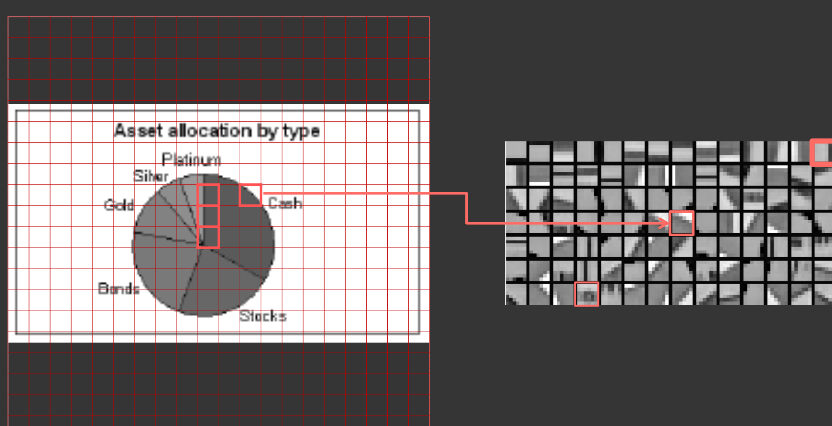
Classifying an Input Image



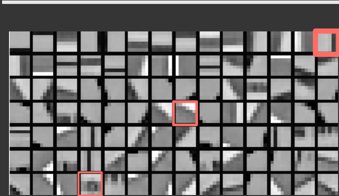
Classifying an Input Image



Classifying an Input Image



Classifying an Input Image

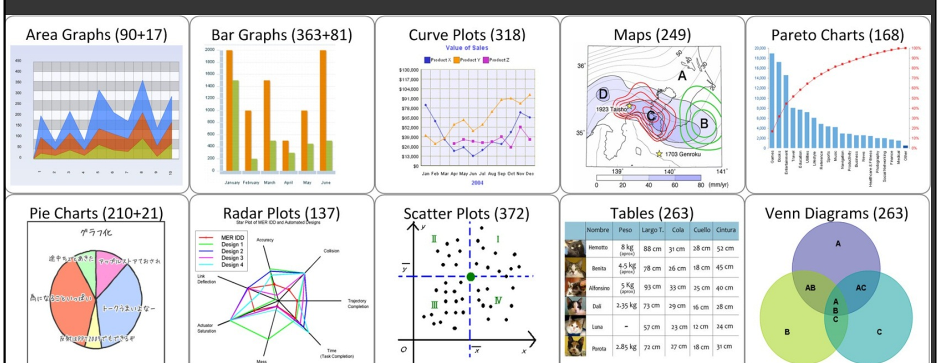


→ **SVM Classifier** → **Pie Chart**

Corpus: 667 charts, 5 chart types [Prasad 2007]	Average Accuracy
[Prasad 2007] Multi-class SVM	84%
ReVision: Multi-class SVM	88%
ReVision: Binary SVM (yes/no for each chart type)	96%

Our Corpus

Over 2500 labeled images and 10 chart types



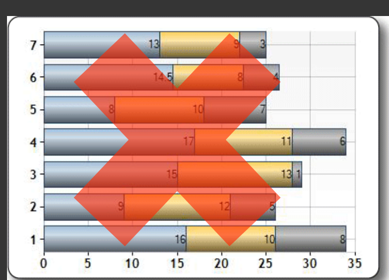
ReVision binary SVMs give 96% classification accuracy

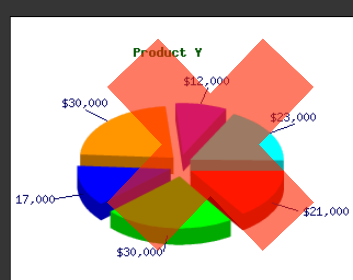
http://vis.berkeley.edu/papers/revision

Mark and Data Extraction

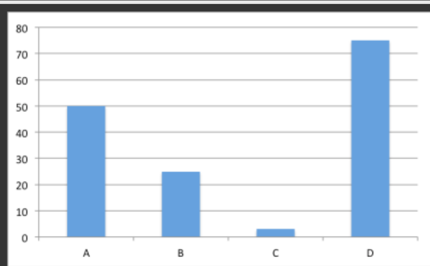
Assumptions

Bar charts and pie charts only
No shading or texture, 3D, stacked bars, or exploded pies

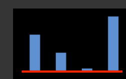




Bar Charts

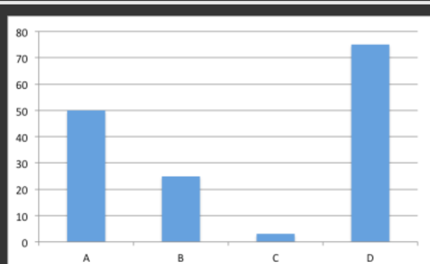


marks: lines

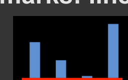


y-value	x-value
50	A
25	B
4	C
75	D

Bar Charts



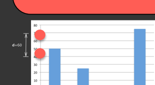
marks: lines



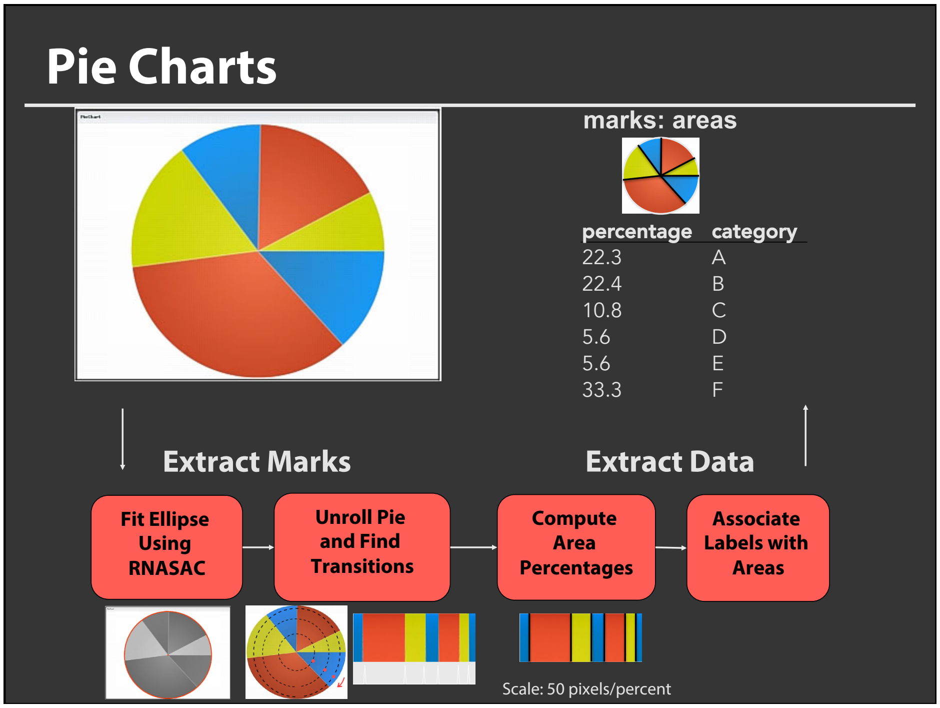
y-value	x-value
50	A
35	B
4	C
75	D

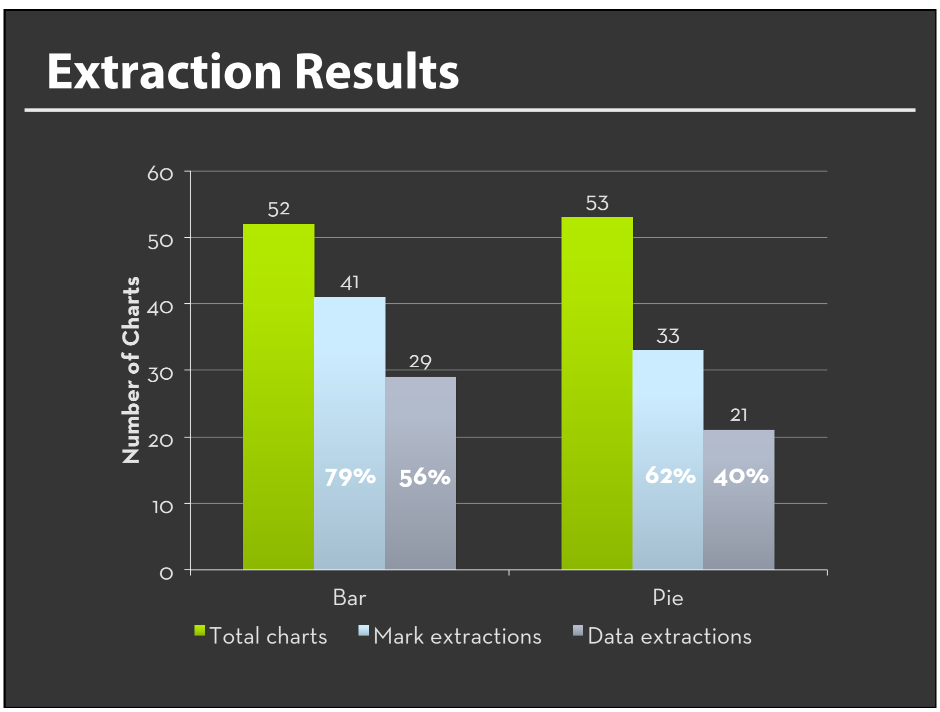
Extract Marks → **Extract Data**

Find Foreground Rectangles → Identify Orientation and Baseline → Recover Bar Values → Associate Labels with Bars



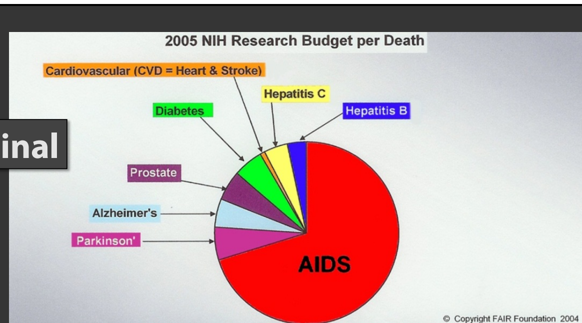
Scale: 2 pixels/unit

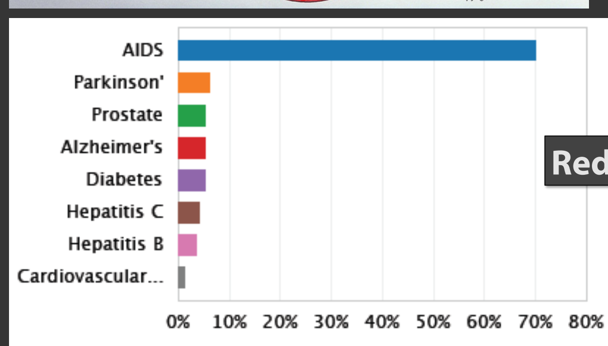

Pie Charts
marks: areas
percentage category
22.3 A
22.4 B
10.8 C
5.6 D
5.6 E
33.3 F
Extract Marks
Extract Data
Fit Ellipse Using RNASAC
Unroll Pie and Find Transitions
Compute Area Percentages
Associate Labels with Areas
Scale: 50 pixels/percent


Extraction Results
Number of Charts
60
50
40
30
20
10
0
52
41
29
79%
56%
53
33
21
62%
40%
Bar
Pie
Total charts
Mark extractions
Data extractions

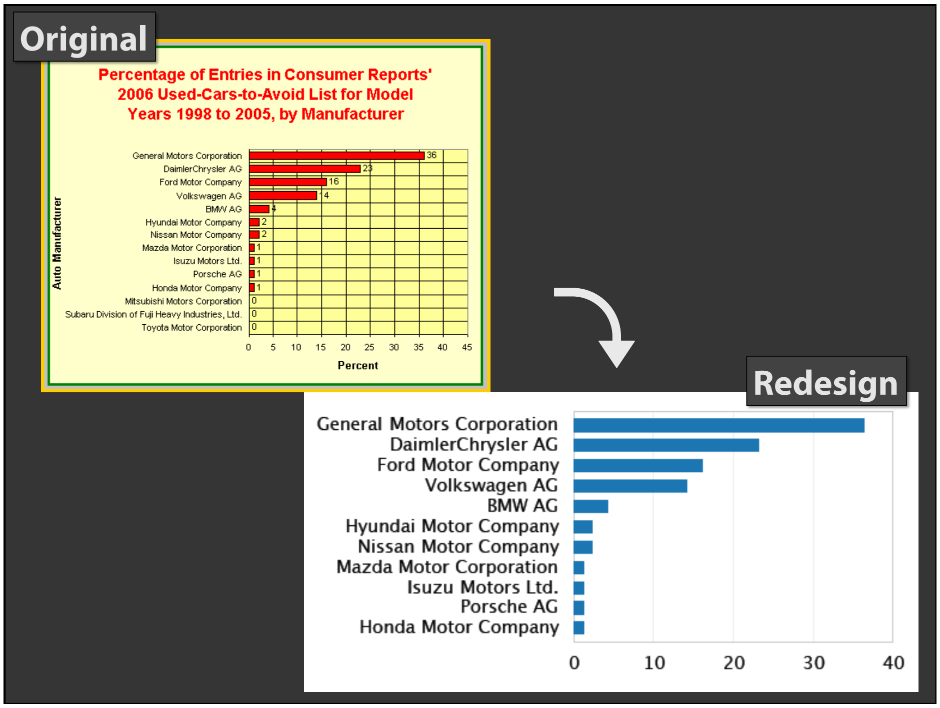
Redesign

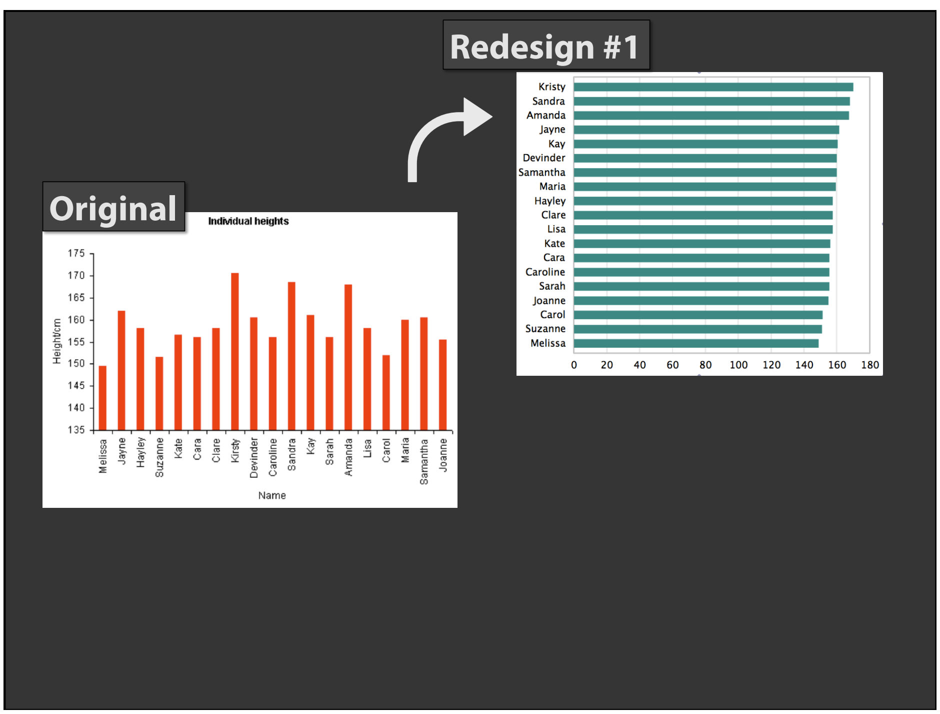
Original

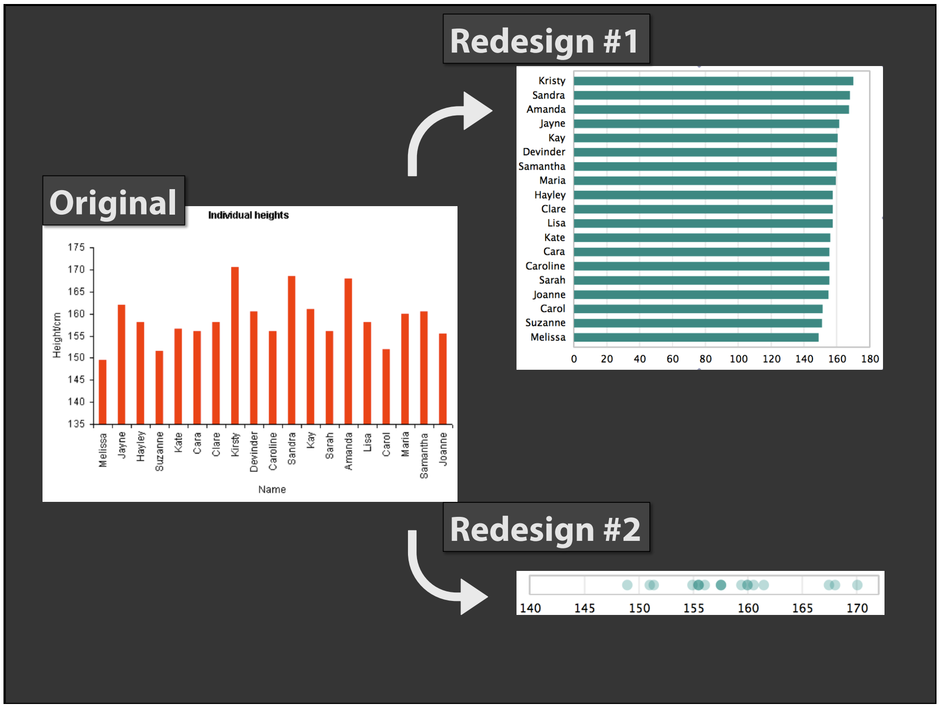


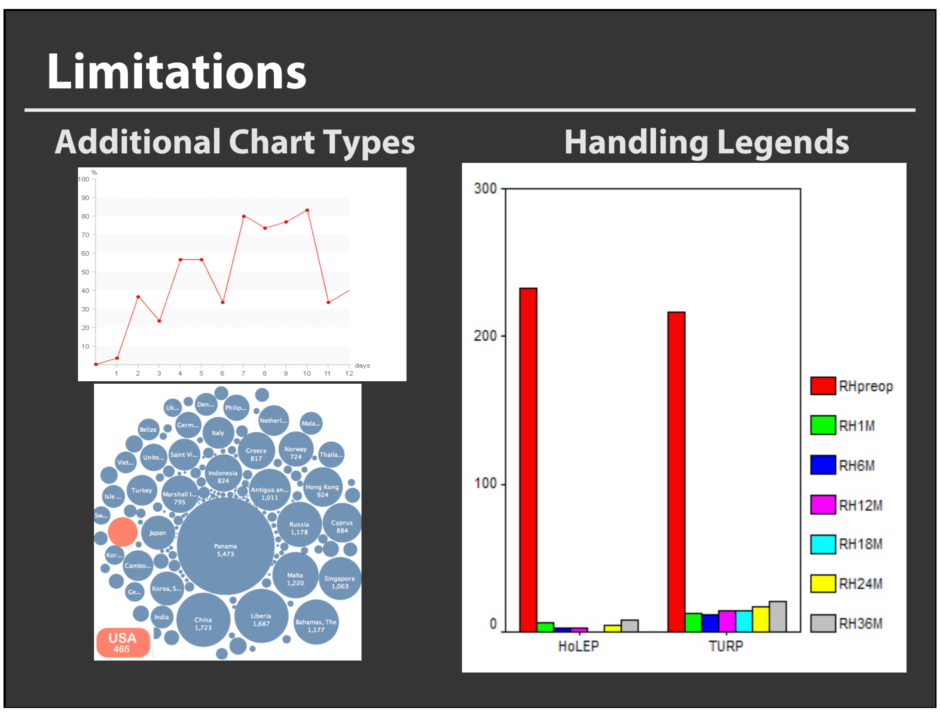


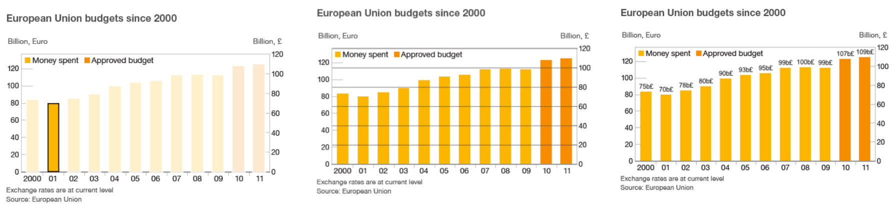
Redesign


Original
Percentage of Entries in Consumer Reports' 2006 Used-Cars-to-Avoid List for Model Years 1998 to 2005, by Manufacturer
Auto Manufacturer
Percent
Redesign
General Motors Corporation
DaimlerChrysler AG
Ford Motor Company
Volkswagen AG
BMW AG
Hyundai Motor Company
Nissan Motor Company
Mazda Motor Corporation
Isuzu Motors Ltd.
Porsche AG
Honda Motor Company


Redesign #1
Original
Individual heights
Height/cm
Name

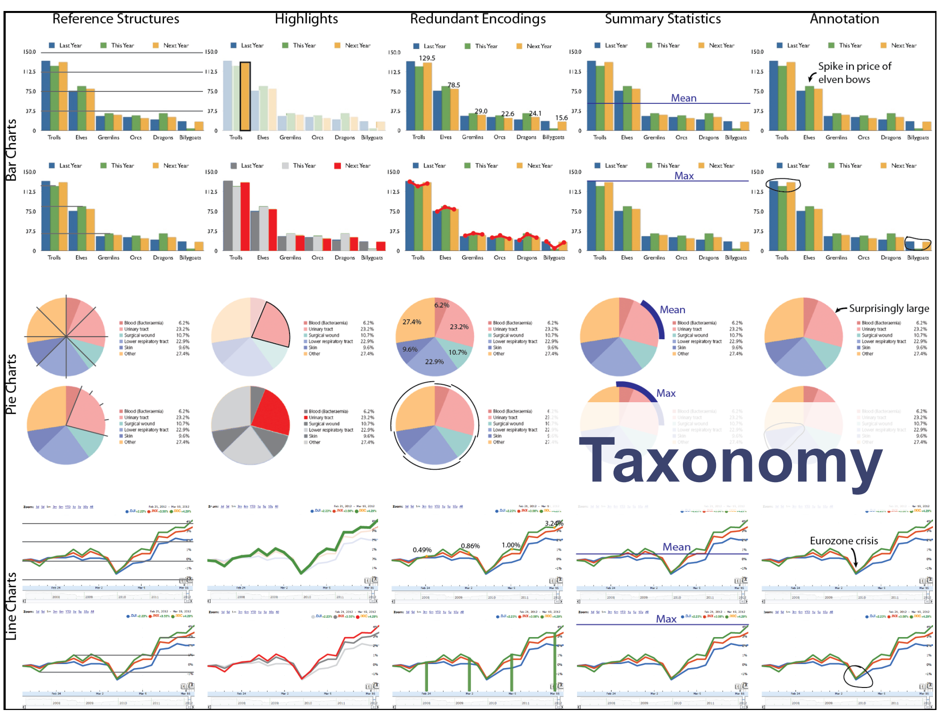

Redesign #1
Kristy
Sandra
Amanda
Jayne
Kay
Devinder
Samantha
Maria
Hayley
Clare
Lisa
Kate
Cara
Caroline
Sarah
Joanne
Carol
Suzanne
Melissa
0 20 40 60 80 100 120 140 160 180
Original
Individual heights
Height/cm
175 170 165 160 155 150 145 140 135
Melissa Jayne Hayley Suzanne Kate Cara Clare Kirsty Devinder Caroline Sandra Kay Sarah Amanda Lisa Carol Maria Samantha Joanne
Name
Redesign #2
140 145 150 155 160 165 170


Limitations
Additional Chart Types
Handling Legends
300
200
100
0
HoLEP
TURP
RHpreop
RH1M
RH6M
RH12M
RH18M
RH24M
RH36M
USA
465



Graphical Overlays

Visual elements that are layered onto a chart to facilitate the perceptual and cognitive processes involved in chart reading



Taxonomy

Graphical overlay gallery

This gallery contains examples of graphical overlays, described in our paper. We have extracted marks and data from the charts using ReVision (for bars and pie charts) and Datathief (for line charts), but all of the overlays are generated in-browser. Try out some of the parameters, or click on an image thumbnail below the gallery to view some example overlays.

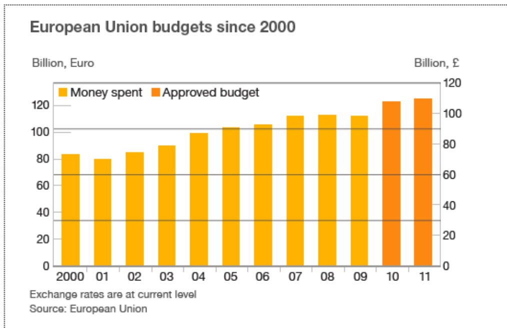


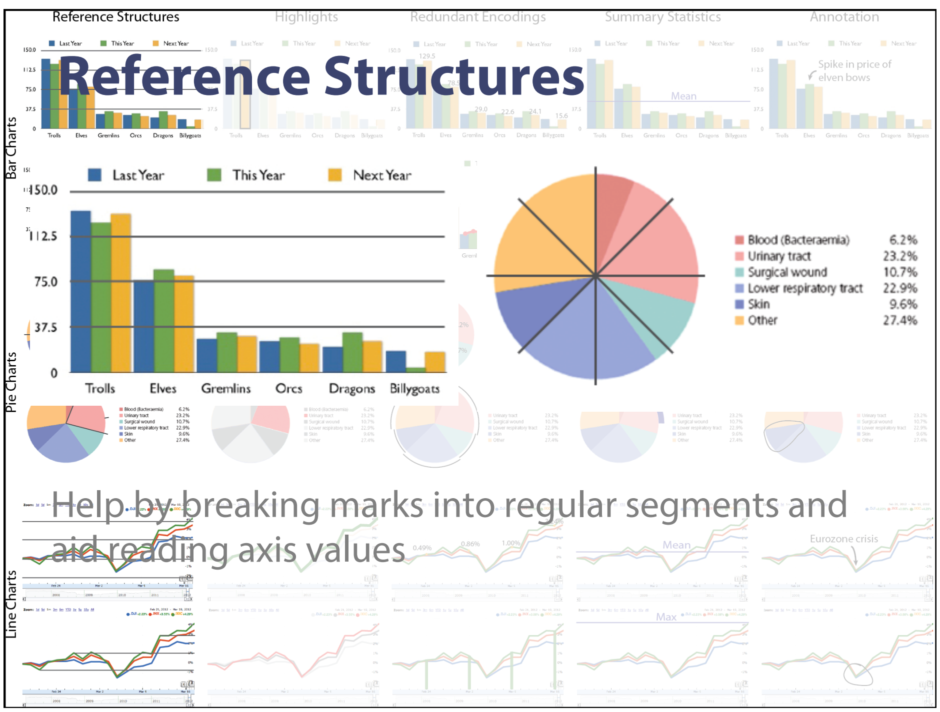
Chart type: Bar
Chart: 00281
Overlay type: Reference structures
Regular gridlines
Lines emanating from marks

Parameters

Overlay Underlay
Static Interactive
Divisions: 4
Line thickness: 1

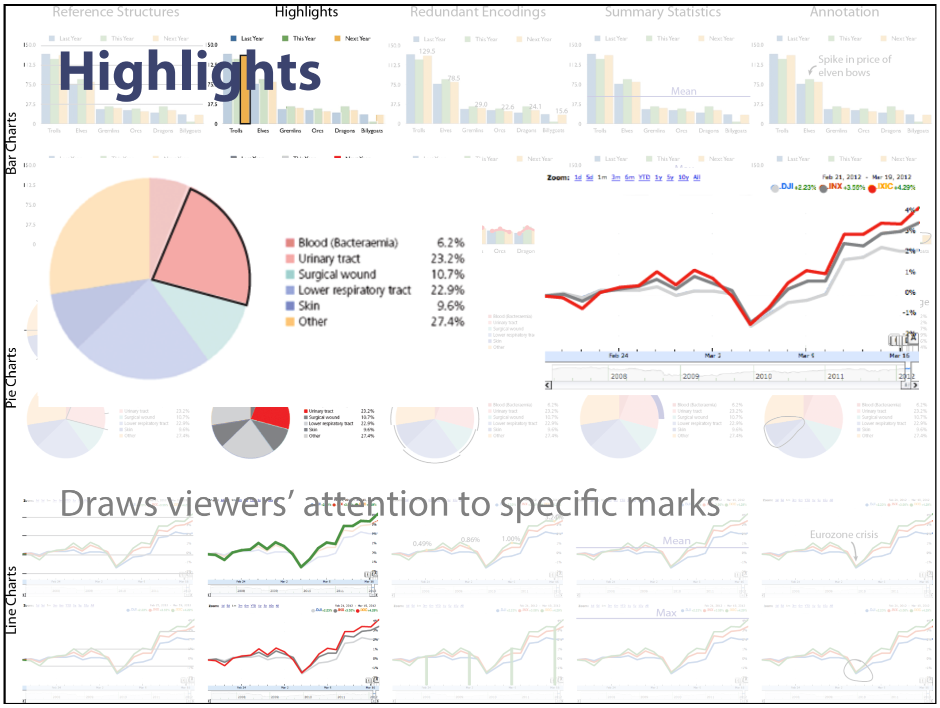
Places regular gridlines at user defined intervals.

Demo

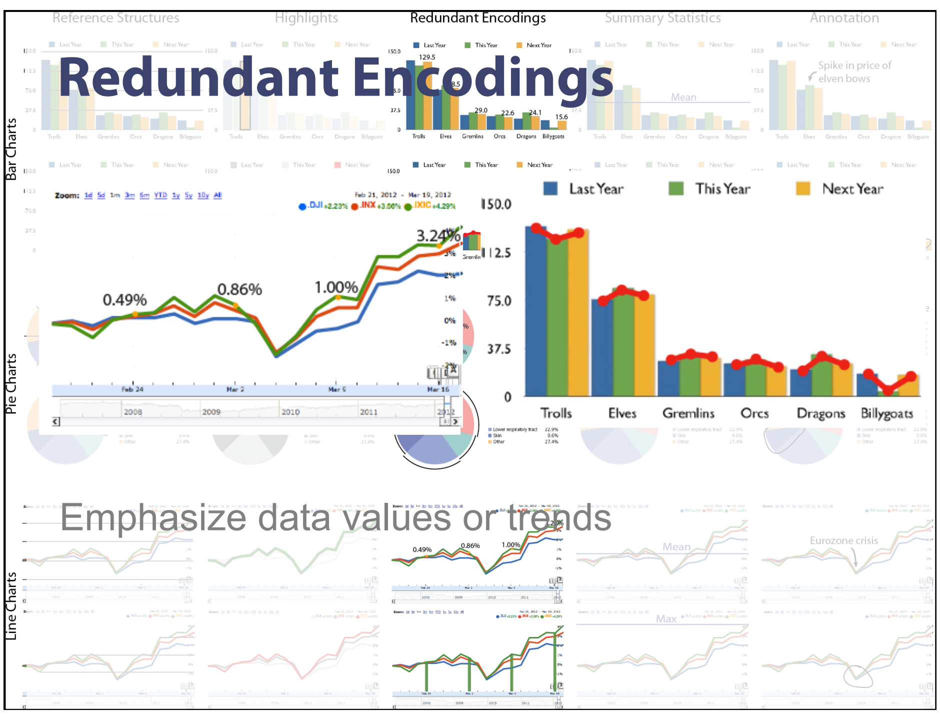


Reference Structures

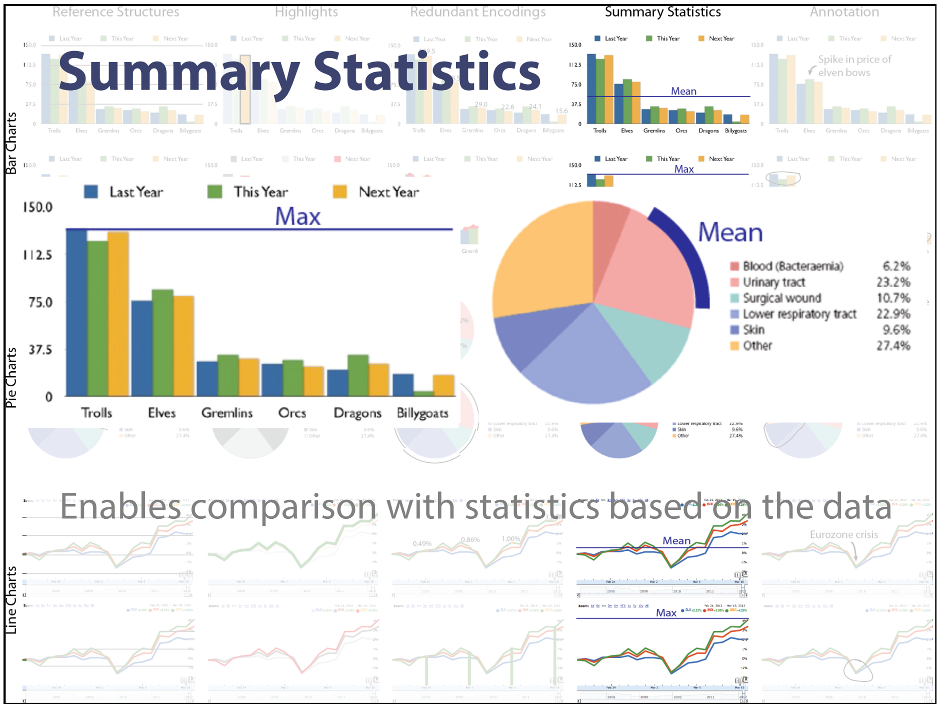
Help by breaking marks into regular segments and aid reading axis values

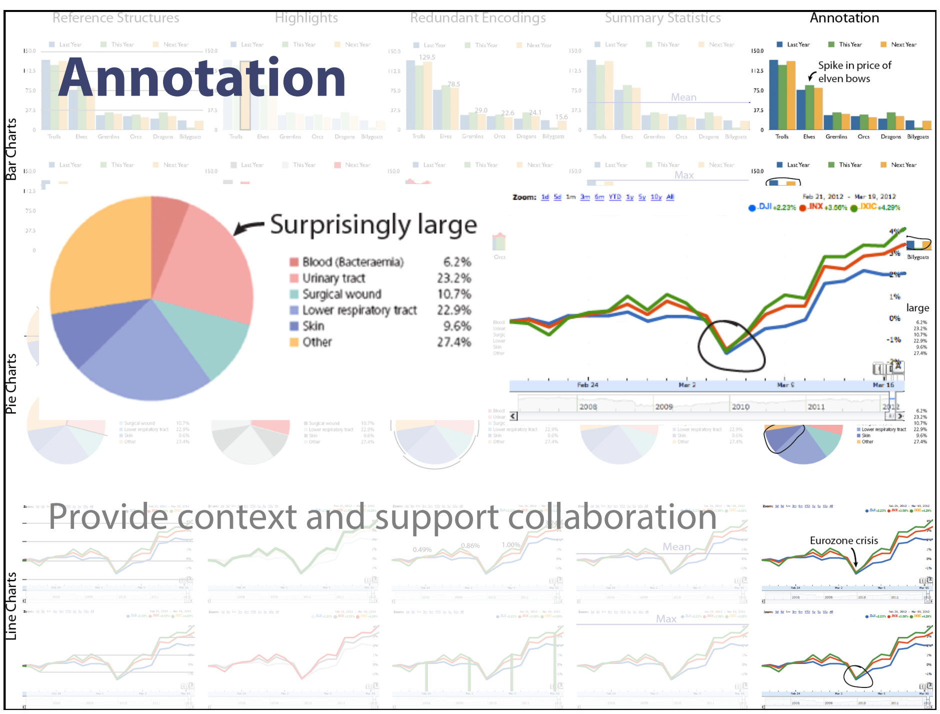


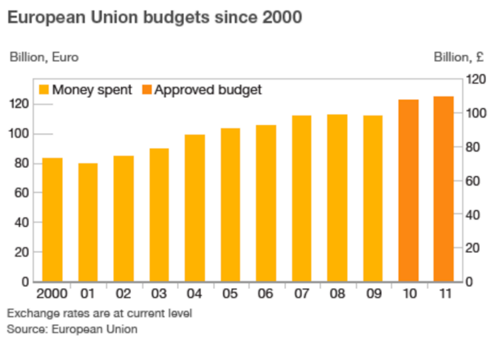
Reference Structures
Highlights
Redundant Encodings
Summary Statistics
Annotation
Highlights
Bar Charts
Pie Charts
Line Charts
Blood (Bacteraemia) 6.2%
Urinary tract 23.2%
Surgical wound 10.7%
Lower respiratory tract 22.9%
Skin 9.6%
Other 27.4%
Spike in price of elven bows
Mean
Eurozone crisis
Max
Draws viewers' attention to specific marks



Reference Structures
Highlights
Redundant Encodings
Summary Statistics
Annotation
Redundant Encodings
Bar Charts
Pie Charts
Line Charts
Last Year
This Year
Next Year
150.0
112.5
75.0
37.5
0
Trolls
Elves
Gremlins
Orcs
Dragons
Billygoats
0.49%
0.86%
1.00%
3.24%
Spike in price of elven bows
Mean
Eurozone crisis
Max
Emphasize data values or trends


Reference Structures
Highlights
Redundant Encodings
Summary Statistics
Annotation
Summary Statistics
Bar Charts
Pie Charts
Line Charts
Last Year
This Year
Next Year
150.0
112.5
75.0
37.5
0
Max
Trolls
Elves
Gremlins
Orcs
Dragons
Billygoats
Mean
Blood (Bacteraemia) 6.2%
Urinary tract 23.2%
Surgical wound 10.7%
Lower respiratory tract 22.9%
Skin 9.6%
Other 27.4%
Enables comparison with statistics based on the data


Reference Structures
Highlights
Redundant Encodings
Summary Statistics
Annotation
Annotation
Bar Charts
Pie Charts
Line Charts
Spike in price of elven bows
Surprisingly large
Blood (Bacteraemia) 6.2%
Urinary tract 23.2%
Surgical wound 10.7%
Lower respiratory tract 22.9%
Skin 9.6%
Other 27.4%
Eurozone crisis
Provide context and support collaboration



mark: lines

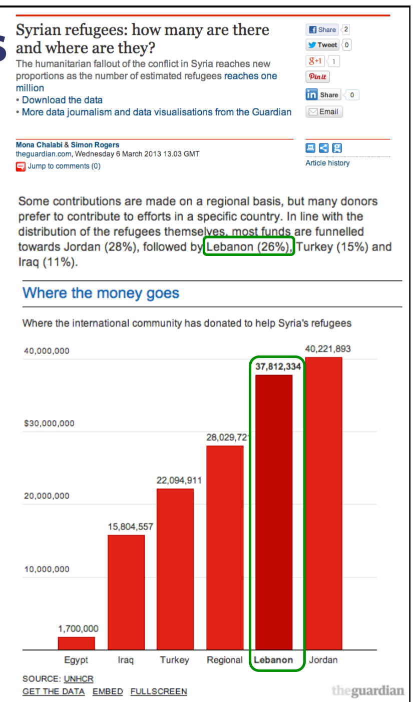
year	money
2000	85
2001	78
2002	87
2003	90
2004	98
…	…

Most overlays only require access to marks

- Reference structures (**marks**)
- Highlights (**marks**)
- Redundant encodings (**marks** and ***data***)
- Summary statistics (**marks**)
- Annotations (**marks**)

Interactive Documents

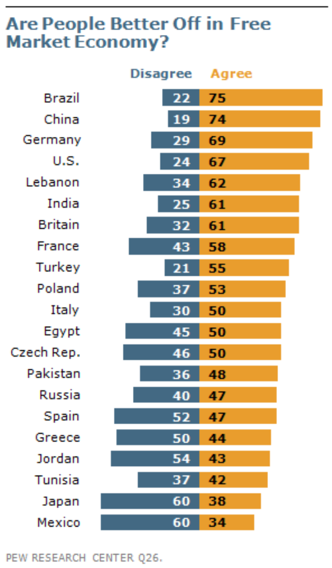
How can we facilitate reading text and charts together?



Goal: Extract references between text and chart

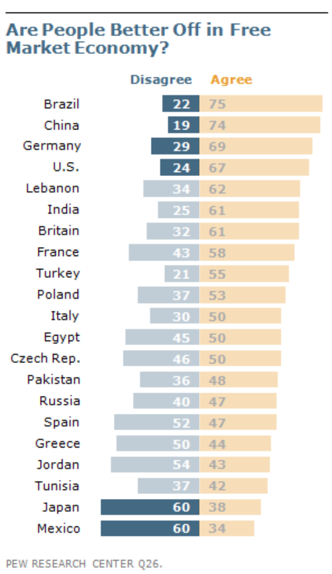
Problem: Diversity of writing styles

Example 1: Pew Research



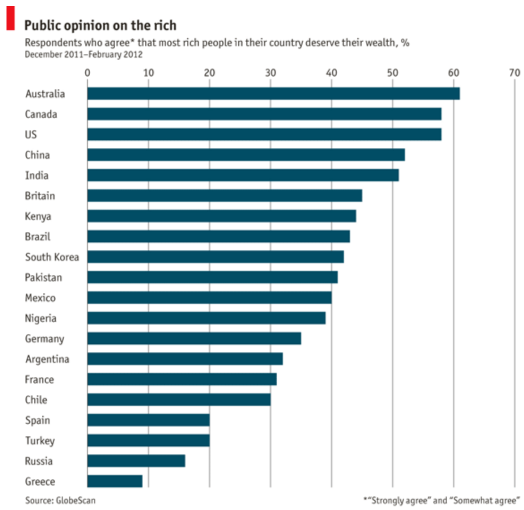
Skepticism for capitalism is lowest in Brazil (22%), China (19%), Germany (29%) (although East Germans are less supportive than West Germans) and the U.S. (24%). Skepticism for free markets is highest in Mexico (60%) and Japan (60%).

Example 1: Pew Research



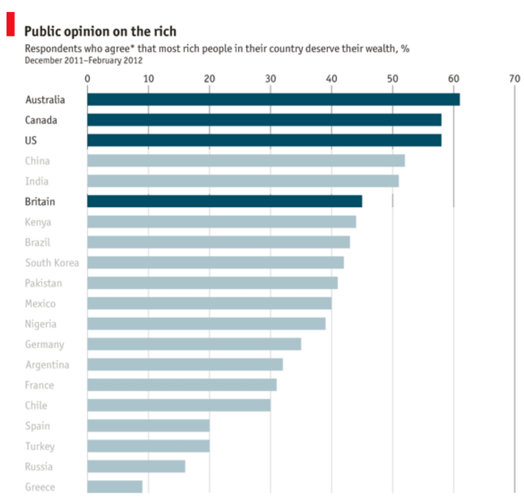
Skepticism for capitalism is lowest in **Brazil (22%), China (19%), Germany (29%)** (although East Germans are less supportive than West Germans) and the **U.S. (24%)**. Skepticism for free markets is highest in **Mexico (60%)** and **Japan (60%).**

Example 2: Economist



Top earners have attracted more opprobrium as their salaries and the performance of the economy have headed in opposite directions. Europeans and Latin Americans tend to have similar attitudes to the rich; the Anglo-Saxon world is a bit more forgiving.

Example 2: Economist



Top earners have attracted more opprobrium as their salaries and the performance of the economy have headed in opposite directions. Europeans and Latin Americans tend to have similar attitudes to the rich; **the Anglo-Saxon world** is a bit more forgiving.

Preprocessing

Document segmentation → Mark and data extraction

Crowdsourcing

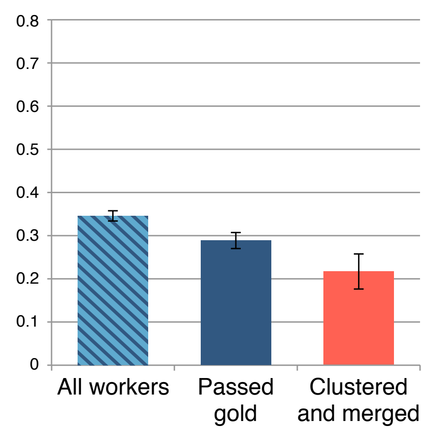
Reference extraction

Clustering and Merging

Merge → Split → Cluster → Select representative

Demo

Evaluation

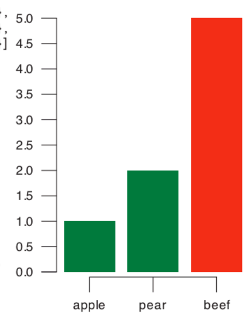


Avg. F_1 distance: expert specified references vs. crowd specified references

Ongoing and Future Work

Deconstructing D3 Charts

```
items = [{name: "apple",type: "fruit", cost: 1.00},
         {name: "pear", type: "fruit", cost: 2.00},
         {name: "beef", type: "meat",  cost: 5.00}]
var bars = svg.selectAll("rect")
                  .data(items)
                  .enter()
                  .append("rect");
bars.attr("x", function(d, i)
        {return i * 25;})
    .attr("y", function(d)
      {return h - d.price * 10;})
    .attr("height", function(d)
      {return d.price * 10;})
    .attr("fill", function(d, i)
      {if(d.type == "fruit"){return "green";}
       else if (d.type == "meat"){return "red";}})
    .attr("width", "20px")
    .attr("stroke-width", 0);
```



Data

deconID	name	type	cost
2	apple	fruit	1.00
3	pear	fruit	2.00
4	beef	meat	5.00

Marks

fill	xPosition	height
green	35 px	20 px
green	60 px	40 px
red	85 px	100 px

Mappings

$type \xrightarrow{\mathcal{C}} fill$
$cost \xrightarrow{\mathcal{L}} height$
$cost \xrightarrow{\mathcal{L}} yPos$
$cost \xrightarrow{\mathcal{L}} area$
$deconID \xrightarrow{\mathcal{L}} xPos$

D3 Code | D3 Chart | Our Deconstruction

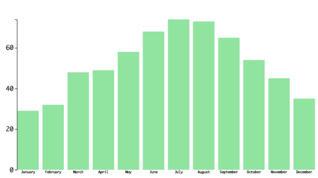
Automatically convert D3 code into mapping based representation to enable redesign and style reuse

Deconstructing and Restyling D3 Visualizations. Jonathan Harper and Maneesh Agrawala. User Interface Software Technology (UIST) 2014.

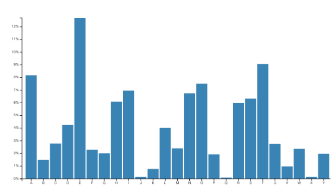
Automatic Redesign

Can we automatically redesign charts to improve

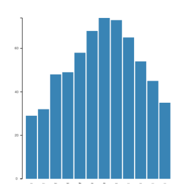
Perceptual effectiveness?
Visual aesthetics?
Accessibility for vision impaired users?



Data Source

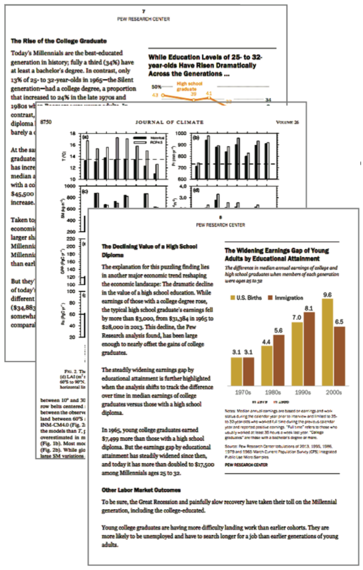


Style Target



Result

Document Collections



Many specialized collections

- Scientific: PLOS, JSTOR, ACM DL, …
- Web visualizations: D3, Processing, …
- News: New York Times, Pew research, …

How can deconstruction aid search?

- Search by chart type, data type, marks, data, …
- Similarity search with inexact matching
- Query expansion

Takeaways

A ***chart*** is a collection of ***mappings between data and marks***

We ***can reconstruct*** this representation ***from chart bitmaps***

Such reconstruction ***enables redesign, reuse*** and ***revitalization***