# Module Master Recherche
# Apprentissage et Fouille

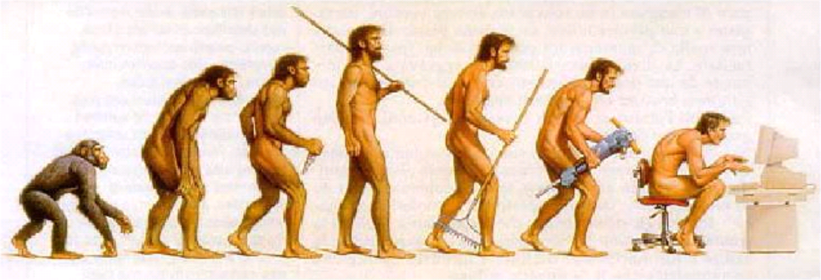
Michele Sebag
http://tao.lri.fr

Automne 2009

# Apprentissage non supervisé

- Case Study
- Data Clustering
- Data Streaming

# Case study: Autonomic Computing



Considering current technologies, we expect that the total number of device administrators will exceed 220 millions by 2010.

Gartner 6/2001

in Autonomic Computing Wshop, ECML / PKDD 2006

Irina Rish & Gerry Tesauro.

# Autonomic Computing

## The need

- Main bottleneck of the deployment of complex systems: shortage of skilled administrators

## Vision

- Computing systems take care of the mundane elements of management by themselves.
- Inspiration: central nervous system (regulating temperature, breathing, and heart rate without conscious thought)

## Goal

Computing systems that manage themselves in accordance with high-level objectives from humans

Kephart & Chess, IEEE Computer 2003

# Autonomic Grid System

- Grid Systems
  Presentation of EGEE, Enabling Grids for e-Science in Europe
- Acquiring the data
  The grid observatory
- Preparation of the data
  - Functional dependencies
  - Dimensionality reduction
  - Propositionalization

# EGEE: Enabling Grids for E-Science in Europe

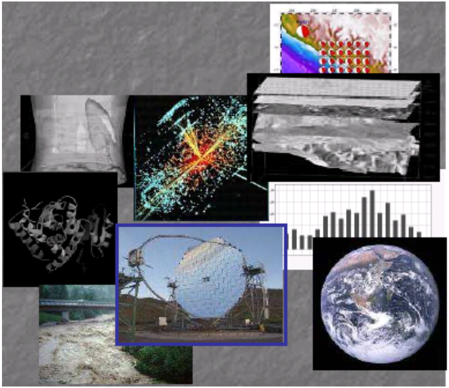
# EGEE, 2

- Infrastructure project started in 2001 $\rightarrow$ FP6 and FP7
- Large scale, production quality grid
- Core node: Lab. Accelerateur Linéaire, Université Paris-Sud
- 240 partners, 41,000 CPUs, all over the world
- 5 Peta bytes storage
- 24 $\times$ 7, 20 K concurrent jobs
- Web: www.eu-egee.org

Storage as important as CPU

# Applications

- High energy physics
- Life sciences
- Astrophysics
- Computational chemistry
- Earth sciences
- Financial simulation
- Fusion
- Multimedia
- Geophysics



# Autonomic Grid

## Requisite: The Grid Observatory

- Cluster in the EGEE-III proposal 2008-2010
- Data collection and publication: filtering, clustering

## Workload management

- Models of the grid dynamics
- Models of requirements and middleware reaction: time series and beyond
- Utility based-scheduling, local and global: MAB problem
- Policy evaluations: very large scale optimization

## Fault detection and diagnosis

- Categorization of failure modes from the Logging and Bookkeeping: feature construction, clustering,
- Abrupt changepoint detection

# Autonomic Grid: The Grid Observatory

## Data acquisition

- Data have not been stored with DM in mind — never
- Data [partially] automatically generated — here
  for EGEE services
  - redundant
  - little expert help

*It's no longer: the expert feeds the machine with data. Rather, machines feed machines...* — *J. Gama*

## Data preprocessing

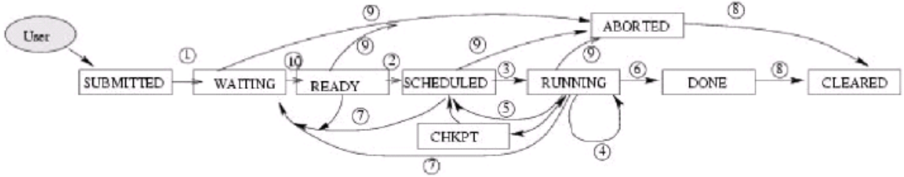
- 80% of the human cost
- Governs the quality of the output

# The grid system and the data

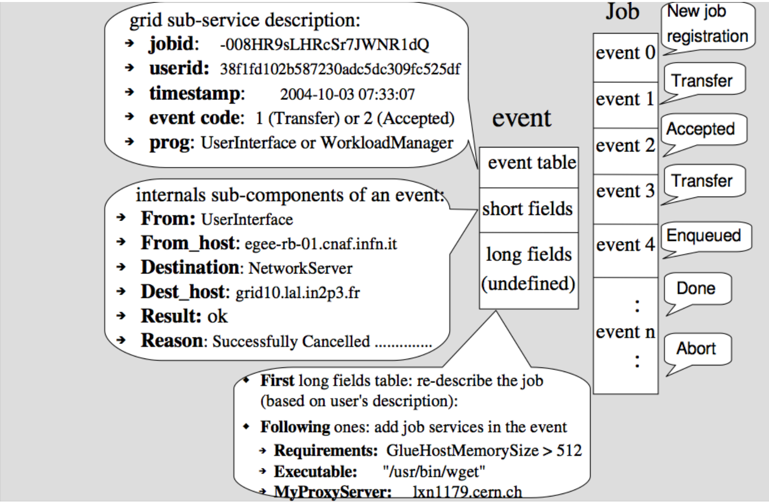
## The Workload Management System

- User Interface — User submits job description and requirements, and gets the results
- Resource Broker — Decides Computing Element
- Job Submission Service — Submits to CE and Checks
- Logging and Bookkeeping Service — Archive the data

## Job Lifecycle



# The data


grid sub-service description:
jobid: -008HR9sLHRcSr7JWNR1dQ
userid: 38f1fd102b587230adc5dc309fc525df
timestamp: 2004-10-03 07:33:07
event code: 1 (Transfer) or 2 (Accepted)
prog: UserInterface or WorkloadManager
internals sub-components of an event:
From: UserInterface
From_host: egee-rb-01.cnaf.infn.it
Destination: NetworkServer
Dest_host: grid10.lal.in2p3.fr
Result: ok
Reason: Successfully Cancelled ..............
event
event table
short fields
long fields (undefined)
First long fields table: re-describe the job (based on user's description):
Following ones: add job services in the event
Requirements: GlueHostMemorySize > 512
Executable: "/usr/bin/wget"
MyProxyServer: lxn1179.cern.ch
Job
event 0
event 1
event 2
event 3
event 4
:
event n
:
New job registration
Transfer
Accepted
Transfer
Enqueued
Done
Abort

# Data Tables

## Events

| jobid | event | code | host | time_stamp | arrived | level |
|---|---|---|---|---|---|---|
| ---BrI1BgbIqkwtszqGfmA | 0 | 17 | atlfarm008.mi.infn.it | 2004-09-17 16:17:48 | 2004-09-17 16:17:49 | 8 |
| ---BrI1BgbIqkwtszqGfmA | 1 | 1 | atlfarm008.mi.infn.it | 2004-09-17 16:17:48 | 2004-09-17 16:17:49 | 8 |
| ---BrI1BgbIqkwtszqGfmA | 2 | 2 | lxb0728.cern.ch | 2004-09-17 16:17:53 | 2004-09-17 16:17:53 | 8 |
| ---BrI1BgbIqkwtszqGfmA | 3 | 4 | lxb0728.cern.ch | 2004-09-17 16:18:00 | 2004-09-17 16:18:01 | 8 |
| ---BrI1BgbIqkwtszqGfmA | 4 | 1 | atlfarm008.mi.infn.it | 2004-09-17 16:18:00 | 2004-09-17 16:18:01 | 8 |
| ---BrI1BgbIqkwtszqGfmA | 5 | 5 | lxb0728.cern.ch | 2004-09-17 16:18:01 | 2004-09-17 16:18:01 | 8 |

## Short Fields

| | | |
|---|---|---|
| 0 | JOBTYPE | SIMPLE |
| 0 | NS | lxb0728.cern.ch:7772 |
| 0 | NSUBJOBS | 0 |
| 0 | SEED | uLUOBArrdV98041PLThJ5Q |
| 0 | SEQCODE | UI=000001:NS=0000000000:WM=000000:BH=0000000000:JSS=000000:LM=000000:LRMS=000000:APP=000000 |
| 0 | SRC_INSTANCE | |
| 1 | DESTINATION | NetworkServer |
| 1 | DEST_HOST | lxb0728.cern.ch |
| 1 | DEST_INSTANCE | lxb0728.cern.ch:7772 |
| 1 | DEST_JOBID | |
| 1 | REASON | |
| 1 | RESULT | START |
| 1 | SEQCODE | UI=000002:NS=0000000000:WM=000000:BH=0000000000:JSS=000000:LM=000000:LRMS=000000:APP=000000 |
| 1 | SRC_INSTANCE | |
| 2 | FROM | UserInterface |
| 2 | FROM_HOST | lxb0728.cern.ch |
| 2 | FROM_INSTANCE | |
| 2 | LOCAL_JOBID | |
| 2 | SEQCODE | UI=000003:NS=0000000001:WM=000000:BH=0000000000:JSS=000000:LM=000000:LRMS=000000:APP=000000 |
| 2 | SRC_INSTANCE | 7772 |
| 3 | QUEUE | /var/edgwl/workload_manager/input.fl |
| 3 | REASON | |
| 3 | RESULT | OK |
| 3 | SEQCODE | UI=000003:NS=0000000003:WM=000000:BH=0000000000:JSS=000000:LM=000000:LRMS=000000:APP=000000 |
| 3 | SRC_INSTANCE | |

# Data Tables

## Long Fields (4Gb)

```
+------------------------+-------+------
| jobid                  | event | name | value
+------------------------+-------+------
| ---BrI1BgbIqkwtszqGfmA |     0 | JDL  |[ requirements = ( ( ( ( Member("VO-atlas-lcg-release
-0.0.2",other.GlueHostApplicationSoftwareRunTimeEnvironment) ) && Member("VO-atlas-release
-8.0.5",other.GlueHostApplicationSoftwareRunTimeEnvironment) ) && ( other.GlueCEPolicyMaxCPUTime >= ( Member("LCG
-2\_1_0",other.GlueHostApplicationSoftwareRunTimeEnvironment) ? ( 36000000 / 60 ) : 36000000 ) / other.GlueHostBenchmarkSI00 ) ) &&
(other.GlueHostNetworkAdapterOutboundIP == true ) ) && ( other.GlueHostMainMemoryRAMSize >= 512 ); RetryCount = 0; edg_jobid =
 "https://lxb0728.cern.ch:9000/---BrI1BgbIqkwtszqGfmA"; Arguments = "dc2.003048.evgen.H4_170_WW._00002.pool.root
dc2.003048.simul.H4_170_WW._00208.pool.root.2 -6 6 50 350 208"; Environment = {
"LEXOR_WRAPPER_LOG=lexor_wrapper.log","LEXOR_STAGEOUT_MAXATTEMPT=5","LEXOR_STAGEOUT_INTERVAL=60",
"LEXOR_LCG_GFAL_INFOSYS=lxb2011.cern.ch:2170","LEXOR_T_RELEASE=8.0.5",
"LEXOR_T_PACKAGE=8.0.5.6/JobTransforms","LEXOR_T_BASEDIR=JobTransforms-08-00-05-06",
"LEXOR_TRANSFORMATION=share/
dc2.g4sim.trf","LEXOR_STAGEIN_LOG=dq_233387_stagein.log","LEXOR_STAGEIN_SCRIPT=dq_233387_stagein.sh",
"LEXOR_STAGEOUT_LOG=dq_233387_stageout.log","LEXOR_STAGEOUT_SCRIPT=dq_233387_stageout.sh" };
 MyProxyServer = "lxb0727.cern.ch"; JobType = "normal"; Executable =
"lexor_wrap.sh"; StdOutput = "dc2.003048.simul.H4_170_WW._00208.job.log.2"; OutputSandbox = {
"metadata.xml","lexor_wrapper.log","dq_233387_stagein.log","dq_233387_stageout.log",
"dc2.003048.simul.H4_170_WW._00208.job.log.2" }; VirtualOrganisation = "atlas";
 rank = ( other.GlueCEStateEstimatedResponseTime > 999 ) ?  -( other.GlueCEStateEstimatedResponseTime ) :  -(
other.GlueCEStateRunningJobs ); Type = "job"; StdError = "dc2.003048.simul.H4_170_WW._00208.job.log.2";
DefaultRank =  -other.GlueCEStateEstimatedResponseTime;
InputSandbox = {
"/home/negri/windmill-0.9.15/lexor/inputsandbox/lexor_wrap.sh",
"/home/negri/windmill-0.9.15/lexor/inputsandbox/dqlcg.py",
"/home/negri/windmill-0.9.15/lexor/inputsandbox/edgrmpi.sh",
"/home/negri/windmill-0.9.15/lexor/inputsandbox/dqrep.pl",
"/home/negri/windmill-0.9.15/lexor/inputsandbox/run_dqlcg.sh","/tmp/lexor/negri/dq_233387_stagein.sh",
"/tmp/lexor/negri/dq_233387_stageout.sh" } ]
+------------------------+-------+------
```

# Apprentissage non supervisé

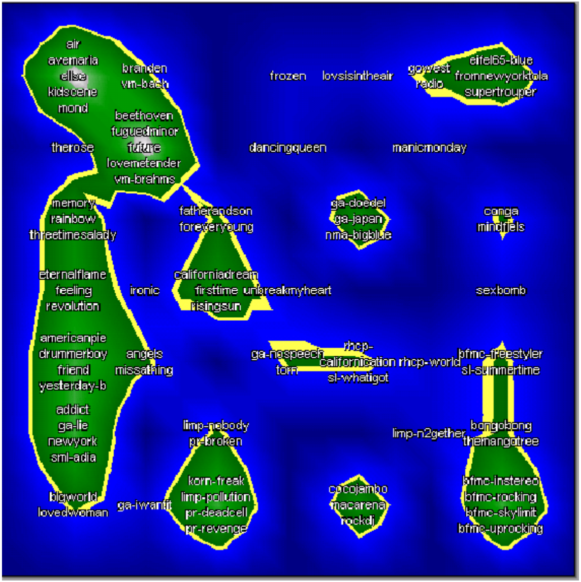
- Case Study
- Data Clustering
- Data Streaming

# Part 1. Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Case study
- Affinity propagation

# Clustering

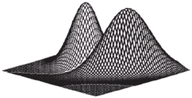
http://www.ofai.at/ elias.pampalk/music/



# Clustering Questions

## Hard or soft ?

- Hard: find a partition of the data
- Soft: estimate the distribution of the data as a mixture of components.



## Parametric *vs* non Parametric ?

- Parametric: number $K$ of clusters is known
- Non-Parametric: find $K$
  (wrapping a parametric clustering algorithm)

## Caveat:

- Complexity
- Outliers
- Validation

# Formal Background

## Notations

| | | |
|---|---|---|
| $\mathcal{E}$ | $\{\mathbf{x}_1, \ldots \mathbf{x}_N\}$ dataset | |
| $N$ | number of data points | |
| $K$ | number of clusters | given or optimized |
| $C_k$ | $k$-th cluster | Hard clustering |
| $\tau(i)$ | index of cluster containing $\mathbf{x}_i$ | |
| $f_k$ | $k$-th model | Soft clustering |
| $\gamma_k(i)$ | $Pr(\mathbf{x}_i \| f_k)$ | |

## Solution

| | |
|---|---|
| Hard Clustering | Partition $\Delta = (C_1, \ldots C_k)$ |
| Soft Clustering | $\forall i \ \sum_k \gamma_k(i) = 1$ |

# Formal Background, 2

## Quality / Cost function

Measures how well the clusters characterize the data

- (log)likelihood — soft clustering
- dispersion — hard clustering

$$\sum_{k=1}^{K} \frac{1}{|C_k|^2} \sum_{\mathbf{x}_i,\mathbf{x}_j \ in \ C_k} d(\mathbf{x}_i, \mathbf{x}_j)^2$$

## Tradeoff

Quality increases with $K$ $\Rightarrow$ Regularization needed

to avoid one cluster per data point

# Clustering vs Classification

Marina Meila
http://videolectures.net/

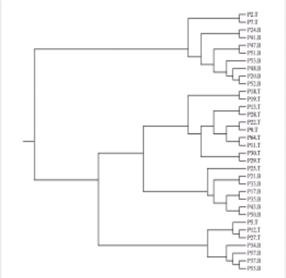
| | Classification | Clustering |
|---|---|---|
| $K$ | # classes (given) | # clusters (unknown) |
| Quality | Generalization error | many cost functions |
| Focus on | Test set | Training set |
| Goal | Prediction | Interpretation |
| Analysis | discriminant | exploratory |
| Field | mature | new |

# Non-Parametric Clustering

## Hierarchical Clustering

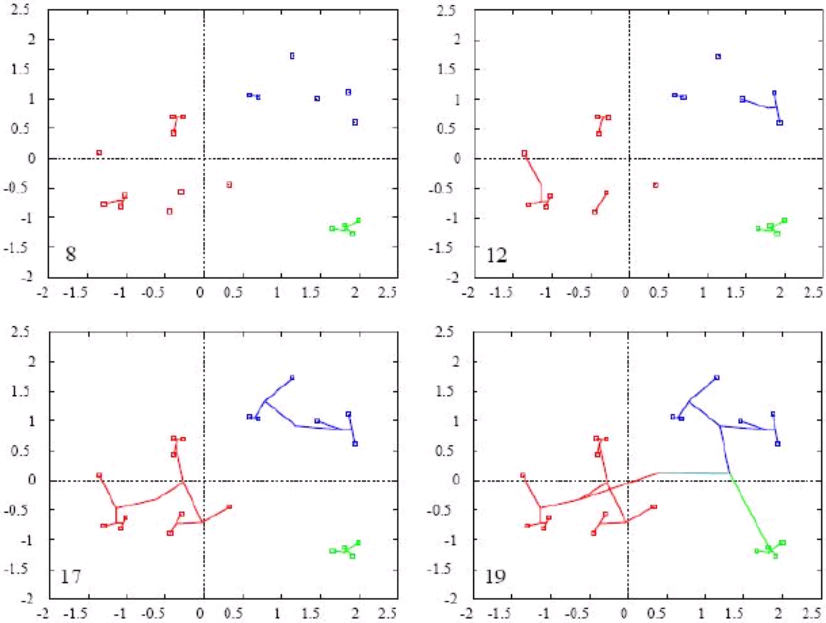
### Principle

- agglomerative (join nearest clusters)
- divisive (split most dispersed cluster)

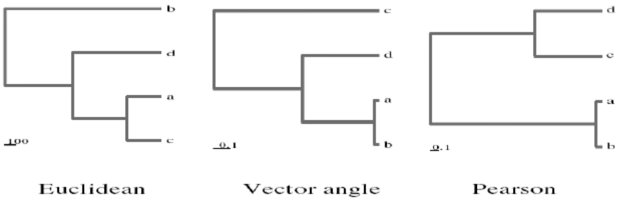


CONS: Complexity $\mathcal{O}(N^3)$

# Hierarchical Clustering, example



# Influence of distance/similarity




$$d(x, x') = \begin{cases} \sqrt{\sum_i (x_i - x_i')^2} & \text{Euclidean distance} \\ 1 - \frac{\sum_i x_i x_i'}{||x||.||x'||} & \text{Cosine angle} \\ 1 - \frac{\sum_i (x_i - \bar{x})(x_i' - \bar{x}')}{||x - \bar{x}||.||x' - \bar{x}'||} & \text{Pearson} \end{cases}$$

# Parametric Clustering

$K$ is known

## Algorithms based on distances

- $K$-means
- graph / cut

## Algorithms based on models

- Mixture of models: EM algorithm

# Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Affinity propagation
- Scalability

# $K$-Means

## Algorithm

1. Init:
   Uniformly draw $K$ points $\mathbf{x}_{i_j}$ in $\mathcal{E}$
   Set $C_j = \{\mathbf{x}_{i_j}\}$
2. Repeat
3. Draw without replacement $\mathbf{x}_i$ from $\mathcal{E}$
4. $\tau(i) = argmin_{k=1 \ldots K}\{d(\mathbf{x}_i, C_k)\}$ find best cluster for $\mathbf{x}_i$
5. $C_{\tau(i)} = C_{\tau(i)} \bigcup \mathbf{x}_i$ add $\mathbf{x}_i$ to $C_{\tau(i)}$
6. Until all points have been drawn
7. If partition $C_1 \ldots C_K$ has changed Stabilize
   Define $\mathbf{x}_{i_k} =$ best point in $C_k$, $C_k = \{x_{i_k}\}$, goto 2.

Algorithm terminates

# $K$-Means, Knobs

| Knob 1 : define $d(\mathbf{x}_i, C_k)$ | favors |
|---|---|
| ▶ $min\{d(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_j \in C_k\}$ | long clusters |
| * $average\{d(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_j \in C_k\}$ | compact clusters |
| ▶ $max\{d(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_j \in C_k\}$ | spheric clusters |

Knob 2 : define "best" in $C_k$

- ▶ Medoid — $argmin_i\{\sum_{\mathbf{x}_j \in C_k} d(\mathbf{x}_i, \mathbf{x}_j)\}$
- * Average (does not belong to $\mathcal{E}$) — $\frac{1}{|C_k|} \sum_{\mathbf{x}_j \in C_k} \mathbf{x}_j$

# No single best choice

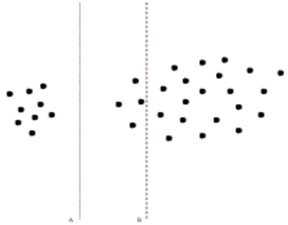



FIG. 1. Optimizing the diameter produces B while A is clearly more desirable.

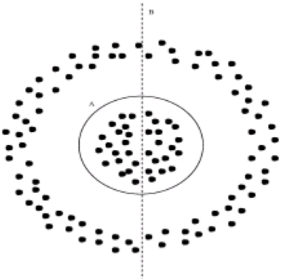



FIG. 2. The inferior clustering B is found by optimizing the 2-median measure.

# $K$-Means, Discussion

PROS

- Complexity $\mathcal{O}(K \times N)$
- Can incorporate prior knowledge initialization

CONS

- Sensitive to initialization
- Sensitive to outliers
- Sensitive to irrelevant attributes

# $K$-Means, Convergence

- For cost function

$$\mathcal{L}(\Delta) = \sum_{k} \sum_{i,j \ / \ \tau(i)=\tau(j)=k} d(\mathbf{x}_i, \mathbf{x}_j)$$

- for $d(\mathbf{x}_i, C_k) =$ average $\{d(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_j \in C_k\}$
- for "best" in $C_k =$ average of $\mathbf{x}_j \in C_k$

$K$-means converges toward a (local) minimum of $\mathcal{L}$.

# $K$-Means, Practicalities

## Initialization

- Uniform sampling
- Average of $\mathcal{E}$ + random perturbations
- Average of $\mathcal{E}$ + orthogonal perturbations
- Extreme points: select $\mathbf{x}_{i_1}$ uniformly in $\mathcal{E}$, then

$$\text{Select } x_{i_j} = argmax\{\sum_{k=1}^{j} d(\mathbf{x}_i, x_{i_k})\}$$

## Pre-processing

- Mean-centering the dataset

# Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Affinity propagation
- Scalability

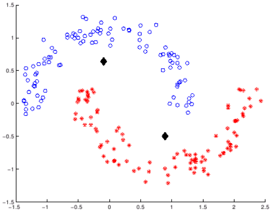
# Model-based clustering

## Mixture of components

- Density $f = \sum_{k=1}^{K} \pi_k f_k$
- $f_k$: the $k$-th component of the mixture
- $\gamma_k(i) = \frac{\pi_k f_k(x)}{f(x)}$
- induces $C_k = \{\mathbf{x}_j \; / \; k = argmax\{\gamma_k(j)\}\}$

## Nature of components: prior knowledge

- Most often Gaussian: $f_k = (\mu_k, \Sigma_k)$
- Beware: clusters are not always Gaussian...



# Model-based clustering, 2

## Search space

- Solution : $(\pi_k, \mu_k, \Sigma_k)_{k=1}^{K} = \theta$

## Criterion: log-likelihood of dataset

$$\ell(\theta) = \log(Pr(\mathcal{E})) = \sum_{i=1}^{N} \log Pr(\mathbf{x}_i) \propto \sum_{i=1}^{N} \sum_{k=1}^{K} \log(\pi_k f_k(\mathbf{x}_i))$$

to be maximized.

# Model-based clustering with EM

## Formalization

- Define $z_{i,k} = 1$ iff $\mathbf{x}_i$ belongs to $C_k$.
- $E[z_{i,k}] = \gamma_k(i)$ prob. $\mathbf{x}_i$ generated by $\pi_k f_k$
- Expectation of log likelihood

$$\begin{aligned} E[\ell(\theta)] &\propto \sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_i(k) \log(\pi_k f_k(\mathbf{x}_i)) \\ &= \sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_i(k) \log \pi_k + \sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_i(k) \log f_k(\mathbf{x}_i) \end{aligned}$$

## EM optimization

**E step** Given $\theta$, compute

$$\gamma_k(i) = \frac{\pi_k f_k(\mathbf{x}_i)}{f(x)}$$

**M step** Given $\gamma_k(i)$, compute

$$\theta^* = (\pi_k, \mu_k, \Sigma_k)^* = argmin E[\ell(\theta)]$$

# Maximization step

$\pi_k$: Fraction of points in $C_k$

$$\pi_k = \frac{1}{N}\sum_{i=1}^{N}\gamma_k(i)$$

$\mu_k$: Mean of $C_k$

$$\mu_k = \frac{\sum_{i=1}^{N}\gamma_k(i)\mathbf{x}_i}{\sum_{i=1}^{N}\gamma_k(i)}$$

$\Sigma_k$: Covariance

$$\Sigma_k = \frac{\sum_{i=1}^{N}\gamma_k(i)(\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)'}{\sum_{i=1}^{N}\gamma_k(i)}$$

# Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Affinity propagation
- Scalability

# Choosing the number of clusters

$K$-means constructs a partition whatever the $K$ value is.

## Selection of K

- Bayesian approaches
  Tradeoff between accuracy / richness of the model
- Stability
  Varying the data should not change the result
- Gap statistics
  Compare with null hypothesis: all data in same cluster.

# Bayesian approaches

## Bayesian Information Criterion

$$BIC(\theta) = \ell(\theta) - \frac{\#\theta}{2} \log N$$

Select $K = \text{argmax } BIC(\theta)$
where $\#\theta$ = number of free parameters in $\theta$:

- if all components have same scalar variance $\sigma$

$$\#\theta = K - 1 + 1 + Kd$$

- if each component has a scalar variance $\sigma_k$

$$\#\theta = K - 1 + K(d + 1)$$

- if each component has a full covariance matrix $\Sigma_k$

$$\#\theta = K - 1 + K(d + d(d - 1)/2)$$

# Gap statistics

## Principle: hypothesis testing

1. Consider hypothesis $H_0$: there is no cluster in the data. $\mathcal{E}$ is generated from a no-cluster distribution $\pi$.
2. Estimate the distribution $f_{0,K}$ of $\mathcal{L}(C_1, \ldots C_K)$ for data generated after $\pi$. Analytically if $\pi$ is simple
   Use Monte-Carlo methods otherwise
3. Reject $H_0$ with confidence $\alpha$ if the probability of generating the true value $\mathcal{L}(C_1, \ldots C_K)$ under $f_{0,K}$ is less than $\alpha$.

Beware: the test is done for all $K$ values...

# Gap statistics, 2

## Algorithm

Assume $\mathcal{E}$ extracted from a no-cluster distribution,
e.g. a single Gaussian.

1. Sample $\mathcal{E}$ according to this distribution
2. Apply $K$-means on this sample
3. Measure the associated loss function

Repeat : compute the average $\bar{\mathcal{L}}_0(K)$ and variance $\sigma_0(K)$
Define the gap:

$$Gap(K) = \bar{\mathcal{L}}_0(K) - \mathcal{L}(C_1, \ldots C_K)$$

Rule Select min $K$ s.t.

$$Gap(K) \geq Gap(K+1) - \sigma_0(K+1)$$

What is nice: also tells if there are no clusters in the data...

# Stability

## Principle

- Consider $\mathcal{E}'$ perturbed from $\mathcal{E}$
- Construct $C'_1, \dots C'_K$ from $\mathcal{E}'$
- Evaluate the "distance" between $(C_1, \dots C_K)$ and $(C'_1, \dots C'_K)$
- If small distance (stability), $K$ is OK

## Distortion $D(\Delta)$

$$\begin{array}{llll} \text{Define} & S & S_{ij} = & < \mathbf{x}_i, \mathbf{x}_j > \\ & & (\lambda_i, v_i) & \text{i-th (eigenvalue, eigenvector) of } S \\ & X & X_{i,j} = & 1 \text{ iff } \mathbf{x}_i \in C_j \end{array}$$

$$D(\Delta) = \sum_i ||\mathbf{x}_i - \mu_{\tau(i)}||^2 = tr(S) - tr(X'SX)$$

Minimal distortion $D^* = tr(S) - \sum_{k=1}^{K-1} \lambda_k$

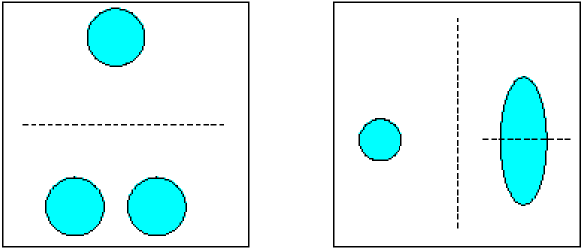
# Stability, 2

## Results

- $\Delta$ has low distortion $\Rightarrow$ $(\mu_1, \ldots \mu_K)$ close to space $(v_1, \ldots v_K)$.
- $\Delta_1$, and $\Delta_2$ have low distortion $\Rightarrow$ "close"
- (and close to "optimal" clustering)

Meila ICML 06

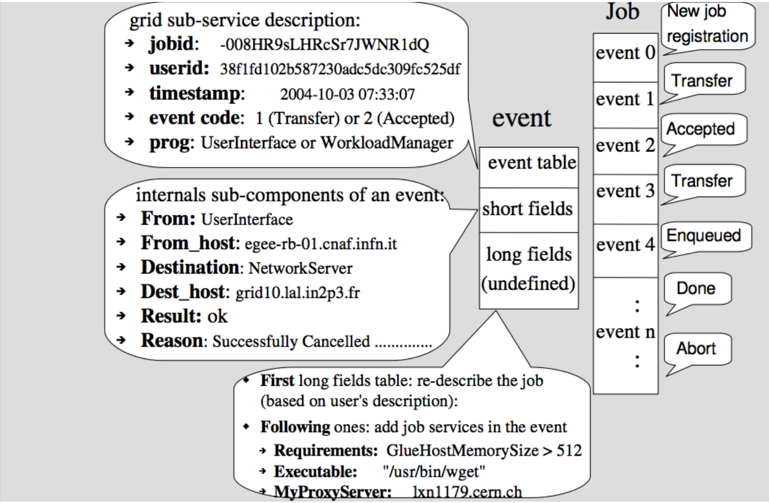
## Counter-example



# Part 1. Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Case study
- Affinity propagation

# Job representation




Xiangliang Zhang et al., ICDM wshop on Data streams, 2007

# Job representation

## Challenges

- Sparse representation, e.g. "user id"
- No natural distance

## Prior knowledge

- Coarse job classification: succeeds (SUC) or fails (FAIL)
- Many failure types: Not Available Resources (NAR); User Aborted (ABU); Generic and non-Generic Error (GNG).
- Jobs are heterogeneous
  - Due to users (advanced or naive)
  - Due to virtual organizations (jobs in physics $\neq$ jobs in biology)
  - Due to time: grid load depends on the community activity

# Feature extraction

## Slicing data

to get rid of heterogeneity

- Split jobs per user: $U_i = \{$ jobs of $i$-th user $\}$
- Split jobs per week: $W_j = \{$ jobs launched in $j$-th week $\}$

## Building features

- Each data slice: a supervised learning problem (discriminating *SUCC* from *FAIL*)

$$h : \mathcal{X} \mapsto \mathbb{R}$$

- Supervised Learning Algorithms:
  - Support Vector Machine SVMLight
  - Optimization of AUC ROGER

# Feature Extraction, 2

## New features

Define

$h_{u,i}$ hypothesis learned from data slice $U_i$

$U : \mathcal{X} \mapsto \mathbb{R}^{\#u}$

$$U(\mathbf{x}) = (h_{u,1}(\mathbf{x}), \ldots h_{u,\#u}(\mathbf{x}))$$

Symmetrically $h_{w,i}$ hypothesis learned from data slice $W_i$

$W : \mathcal{X} \mapsto \mathbb{R}^{\#w}$

$$W(\mathbf{x}) = (h_{w,1}(\mathbf{x}), \ldots h_{w,\#w}(\mathbf{x}))$$

## Change of representation

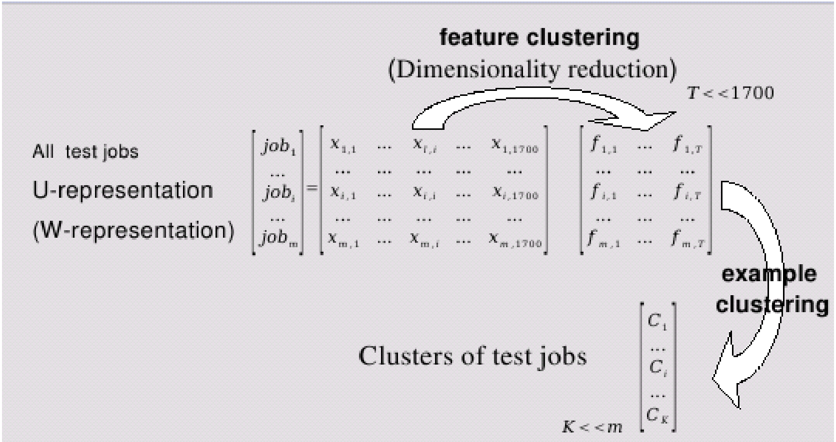
$$\begin{array}{ccl} \mathcal{E} & \rightarrow & \mathcal{E}_U = \{(U(\mathbf{x}_i), y_i), i = 1 \ldots N\} \\ & \rightarrow & \mathcal{E}_W = \{(W(\mathbf{x}_i), y_i), i = 1 \ldots N\} \end{array}$$

## Discussion

- Natural distance on $\mathbb{R}^d$
- But new attributes $h_{u,i}$ likely to be redundant

# Feature Extraction: Double clustering

Slonim & Tishby, 2000



# Experimental setting

## The datasets

- Training set $\mathcal{E}$: 222,500 jobs 36% SUCC, 74% FAIL
- Test set $\mathcal{T}$: 21,512 jobs

## Hypothesis construction

- SVM: one hypothesis per slice: $U : \mathcal{X} \mapsto \mathbb{R}^{34}$, $W : \mathcal{X} \mapsto \mathbb{R}^{45}$
- ROGER: 50 hypotheses per slice $U : \mathcal{X} \mapsto \mathbb{R}^{1700}$, $W : \mathcal{X} \mapsto \mathbb{R}^{2250}$

## Clustering

Foreach $K = 5 \ldots 30$, Apply $K$-means to $\mathcal{T}$

- Considering new representations $U$ and $W$
- Learned after SVM and Roger.

# Goal of Experiments

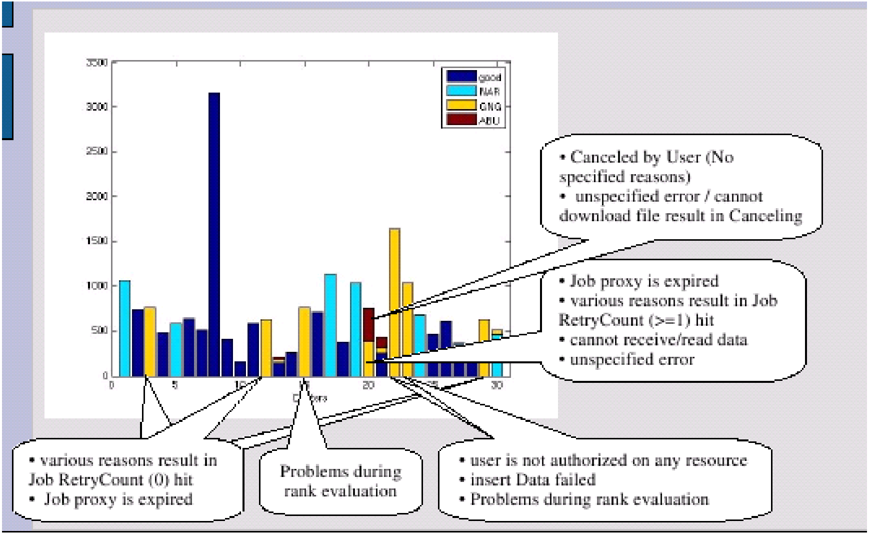
### Interpretation

Examine the clusters

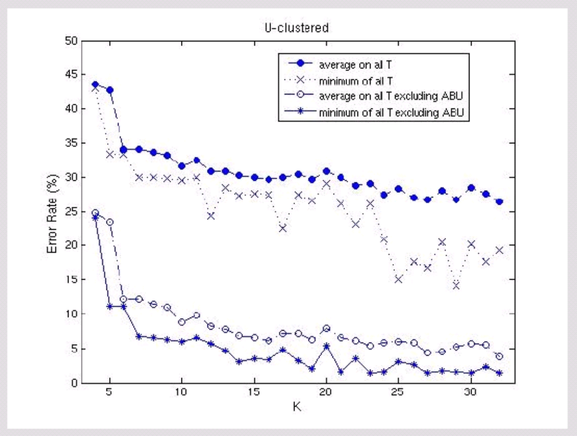
### Stability

- Compare $\Delta_K$ and $\Delta_{K'}$
- Compare $\Delta_{K,U}$ and $\Delta_{K,W}$

# Interpretation



# Interpretation, 2



# Interpretation, 3

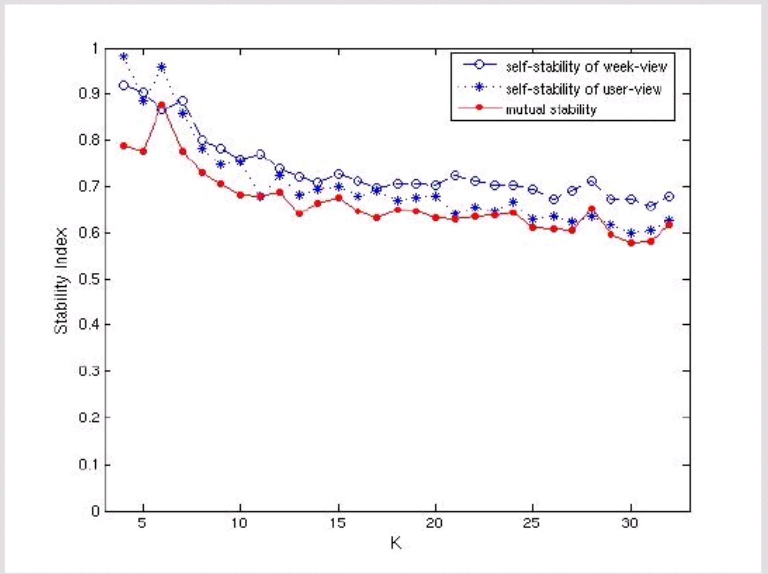
## Pure clusters

- Most clusters are pure wrt sub-classes NAR, GNG
  which were unknown from the algorithm
- Finer-grained classes are discovered: Problem during rank evaluation; job proxy expired; insert Data failed
- ABU class (1.2%) is not properly identified:
  many reasons why job might be *Aborted by User*

## Usage

Use prediction for user-friendly service
Anticipate job failures

# Stability



self-stability of week-view
self-stability of user-view
mutual stability
Stability Index
K
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
5
10
15
20
25
30

# Stability, 2

- Stability wrt initialization, for both $W$ and $U$ representations
- Stability of clusters based on $W$ and $U$-based representations
- Decreases gracefully with $K$
  (optimal value $= 1$)

# Grid Modelling, wrap-up

## Conclusion

- Importance of representation as usual
- Clustering: stable wrt $K$ and representation change
  re-discovers types of failures
  discovers finer-grained failures

## Future work

- Cluster users (= sets of jobs)
- Cluster weeks (= sets of jobs)
- Find scenarios
  naive users gaining expertise;
  grid load & temporal regularities
- Identify communities of users.
- Use scenarios to test/optimize grid services (e.g. scheduler)

# Autonomic Computing, wrap-up

## Huge needs

- Modelling systems
  Black box to calibrate, train, optimize services
- Understanding systems — Hints to repair, re-design systems

## Dealing with Complex Systems

- Findings often challenge conventional wisdom
- Theoretical *vs* Empirical models
- Complex systems are counter-intuitive — sometimes

# Autonomic Computing, wrap-up, 2

## Good practice

- No Magic !
  *I don't see anything, I'll use ML or DM*
- Use all of your prior knowledge
  *If you can measure/model it, don't guess it!*
- Have conjectures
- Test them!

Beware: False Discovery Rate

# Part 1. Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Case study
- Affinity propagation

# From K-Means to K-Centers

## Assumptions for K-Means

- A distance or dissimilarity
- Possibility to create artefacts — barycenters
- Not applicable in some domains — *average molecule?* *average sentence?*

## K-Centers, position of the problem

- A combinatorial optimization problem.
  Find $\sigma : \{1, \ldots, N\} \mapsto \{1, \ldots, N\}$ minimizing:

$$E[\sigma] = \sum_{i=1}^{N} d(\mathbf{x}_i, \mathbf{x}_{\sigma(i)})$$

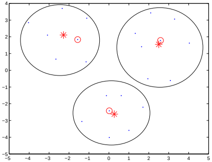
*(What is missing here ?)*

# Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Affinity propagation
- Scalability

# Motivations

Clustering: Unsupervised learning



Affinity Propagation and State of the art

| | **K-means** | **K-centers** | **AP** |
|---|---|---|---|
| exemplar | artefact | actual point | actual point |
| parameter | K | K | $s^*$ (penalty) |
| algorithm | greedy search | greedy search | message passing |
| performance | not stable | not stable | stable |
| complexity | $N \times K$ | $N \times K$ | $N^2 log(N)$ |

Clustering by Passing Messages Between Data Points. B.J. Frey, D. Dueck. Science 2007

# Affinity Propagation

Given

$\mathcal{E} = \{e_1, e_2, ..., e_N\}$ *elements*

$d(e_i, e_j)$ *their dissimilarity*

Find $\sigma : \mathcal{E} \mapsto \mathcal{E}$ $\sigma(e_i)$, exemplar representing $e_i$

such that:

$$\sigma = argmax \sum_{i=1}^{N} S(e_i, \sigma(e_i))$$

where $\begin{cases} S(e_i, e_j) = -d^2(e_i, e_j) & \text{if } i \neq j \\ S(e_i, e_i) = -s^* \end{cases}$ $s^*$: penalty parameter

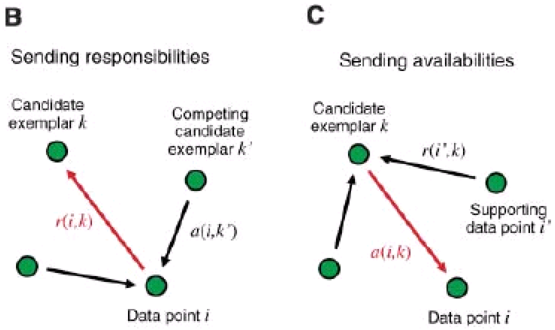
Particular cases

- $s^* = \infty$, only one exemplar 1 cluster
- $s^* = 0$, every point is an exemplar N clusters

# Affinity Propagation, Principle

## Algorithm: Message propagation

- Responsibility $r(i,k)$ could $\mathbf{x}_k$ be examplar for $\mathbf{x}_i$
- Availability $a(i,k)$.



# Affinity Propagation, 2

Two types of messages

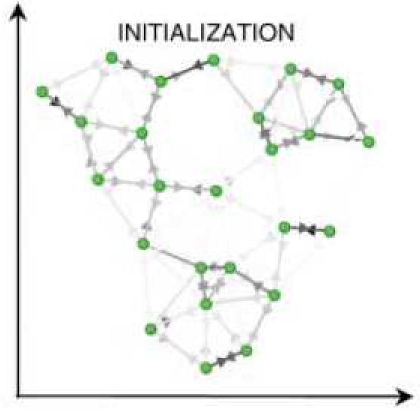
- $r(i,k)$ : Responsibility of $i$ to $k$
- $a(i,k)$ : Availability of $i$ as examplar for $k$

Rules of propagation

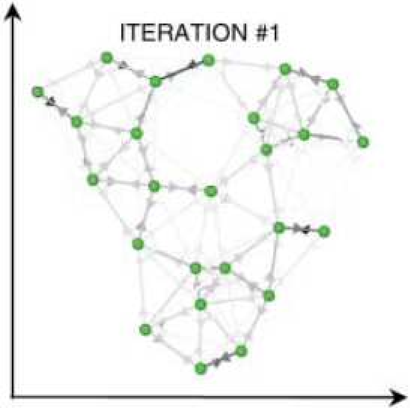
$$
\begin{aligned}
r(i,k) &= S(e_i, e_k) - \max_{k', k' \neq k}\{a(i,k') + S(e_i, e'_k)\} \\
r(k,k) &= S(e_k, e_k) - \max_{k', k' \neq k}\{S(e_k, e'_k)\}
\end{aligned}
$$

$$
\begin{aligned}
a(i,k) &= min\,\{0, r(k,k) + \textstyle\sum_{i', i' \neq i,k} \max\{0, r(i',k)\}\} \\
a(k,k) &= \textstyle\sum_{i', i' \neq k} \max\{0, r(i',k)\}
\end{aligned}
$$

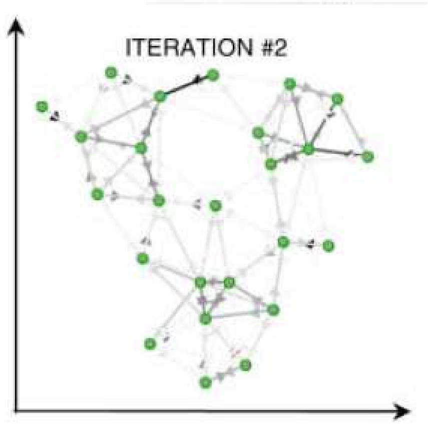
# Iterations of Message passing



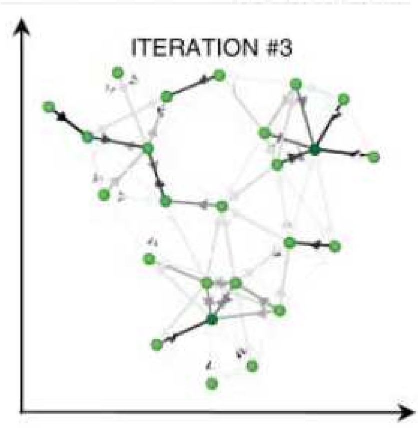
# Iterations of Message passing



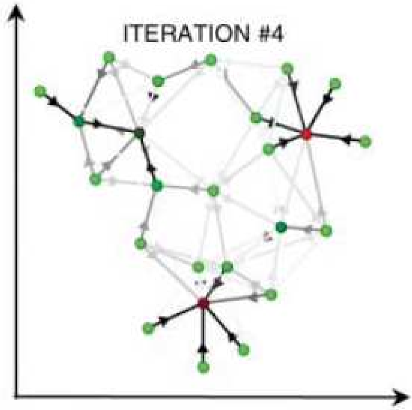
# Iterations of Message passing



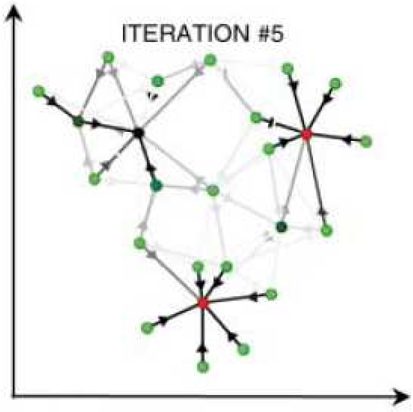
# Iterations of Message passing



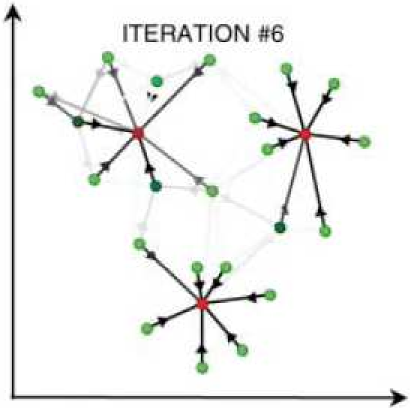
# Iterations of Message passing



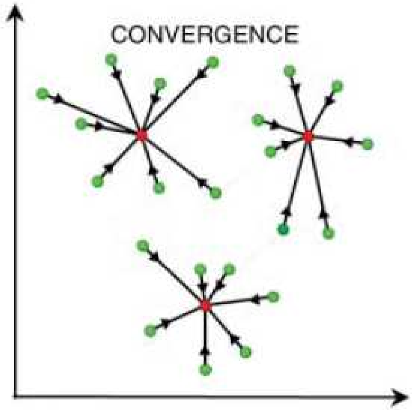
# Iterations of Message passing



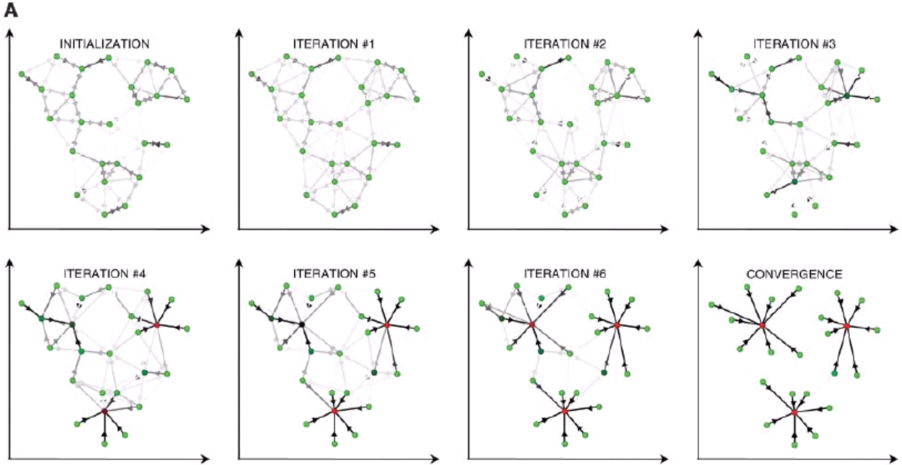
# Iterations of Message passing



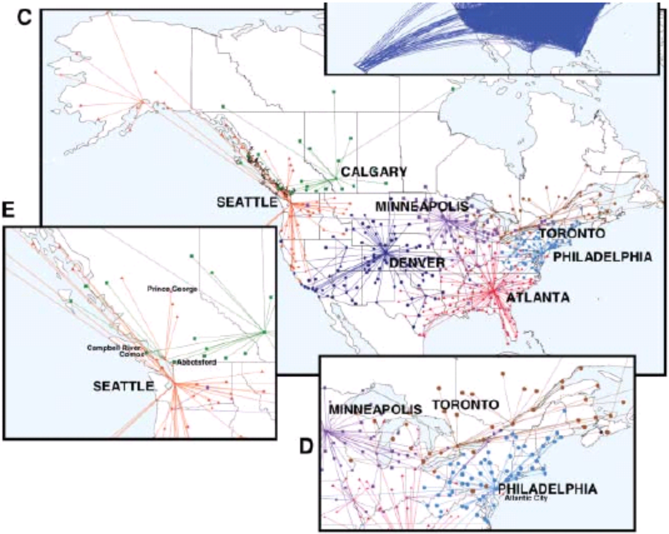
# Iterations of Message passing



# Affinity Propagation, cont'd



# Affinity Propagation, cont'd



# Affinity Propagation in a Nutshell

WHEN to use it ?
When averages don't make sense e.g., molecules; documents

PROS vs $K$-centers
Lower distortion $D([\sigma]) = \sum_{i=1}^{N} d^2(e_i, \sigma(e_i))$

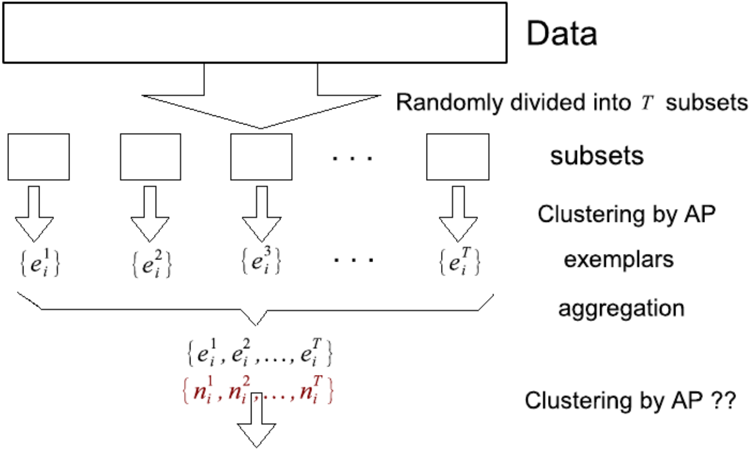
CONS: Computational complexity

- Similarity computation: $\mathcal{O}(N^2)$
- Message passing: $\mathcal{O}(N^2 \log N)$

# Clustering

- K-Means
- Expectation Maximization
- Selecting the number of clusters
- Affinity propagation
- Scalability

# Hierarchical AP




**Clustering data streams: Theory and practice. S. Guha, A. Meyerson, N. Mishra, R. Motwani. TKDE 2003.**

# Weighted AP

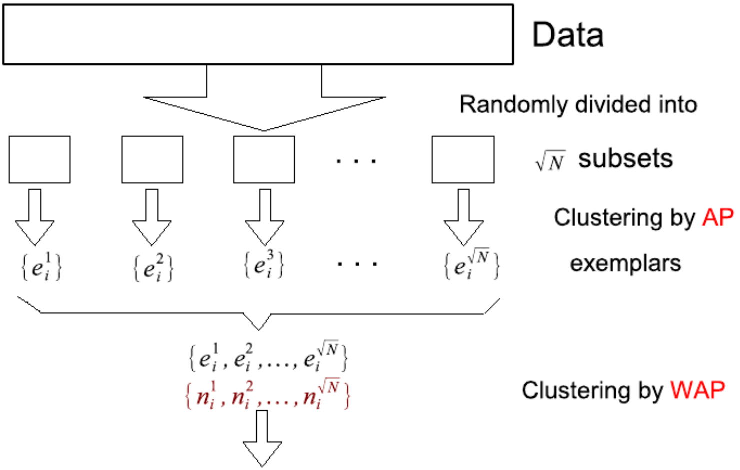
| AP | | WAP |
|---|---|---|
| $e_i$ | | $(e_i, n_i)$ |
| $S(e_i, e_j)$ | $\rightarrow$ | $n_i \times S(e_i, e_j)$ |
| $S(e_i, e_i)$ | $\rightarrow$ | $S(e_i, e_i) + (n_i - 1) \times \epsilon$ |

With $S(e_i, e_j)$ price for $e_i$ to select $e_j$ as an exemplar
$\epsilon$ variance of $n_i$ points

**Proposition**

$$WAP \equiv AP \text{ with duplicated elements}$$

# Hierarchical WAP




- Complexity of HiWAP is $\mathcal{O}(N^{3/2})$
- $\longrightarrow$ can be iteratively reduced to $\mathcal{O}(N^{1+\gamma})$

# Apprentissage non supervisé

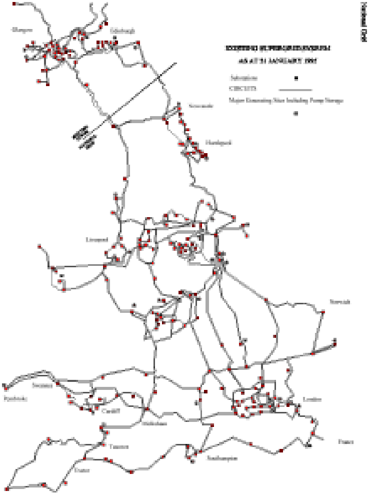
- Case Study
- Data Clustering
- Data Streaming

# Part 2. Data Streaming

- When: data, specificities
- What: goals
- How: algorithms

More: see Joao Gama's tutorial,
http://wiki.kdubiq.org/summerschool2008/index.php/Main/Materials

# Motivations



Electric Power Network

# Data

Input

- Continuous flow of (possibly corrupted) data, high speed
- Huge number of sensors, variable along time (failures)
- Spatio-temporal data

Output

- Cluster: profiles of consumers
- Prediction: peaks of demand
- Monitor Evolution: Change detection, anomaly detection

# Where is the problem ?

## Standard Data Analysis

- Select a sample
- Generate a model (clustering, neural nets, ...)

# Where is the problem ?

## Standard Data Analysis

- Select a sample
- Generate a model (clustering, neural nets, ...)

## Does not work...

- World is not static
- Options, Users, Climate, ... change

# Specificities of data

Domain

- Radar: meteorological observations
- Satellite: images, radiation
- Astronomical surveys: radio
- Internet: traffic logs, user queries, ...
- Sensor networks
- Telecommunications

Features

- Most data never seen by humans
- Need for REAL-TIME monitoring, (intrusion, outliers, anomalies,,,)

NB: Beyond ML scope: data are not iid (independent identically distributed)

# Data streaming Challenges

Maintain Decision Models in real-time

- incorporate new information                comply with speed
- forget old/outdated information
- detect changes and adapt models accordingly

Unbounded training sets                Prefer fast approximate answers...

- Approximation: Find answer with factor $1 \pm \epsilon$
- Probably correct: Pr(answer correct ) $= 1$ -$\delta$
- PAC: $\epsilon, \delta$ (Probably Approximately Correct)
- Space $\approx \mathcal{O}(1/\epsilon^2 log(1/\delta))$

# Data Mining vs Data Streaming

| | **Traditional** | **Stream** |
|---|---|---|
| **Nr. of Passes** | Multiple | Single |
| **Processing Time** | Unlimited | Restricted |
| **Memory Usage** | Unlimited | Restricted |
| **Type of Result** | Accurate | Approximate |
| **Distributed** | No | Yes |

# What: queries on a data stream

- Sample
- Count number of distinct values / attribute
- Estimate sliding average (number of 1's in a sliding window)
- Get top-k elements

Application: Compute entropy of the stream

$$H(x) = \sum p_i log_2(p_i)$$

useful to detect anomalies

# Sampling

Uniform sampling: each one out of $n$ examples is sampled with probability 1/n.
What if we don't know the size ?

Standard

- Sample instances at periodic time intervals
- Loss of information

Reservoir Sampling

- Create buffer size $k$
- Insert first $k$ elements
- Insert $i$-th element with probability $k/i$
- Delete a buffer element at random

Limitations

- Unlikely to detect changes/anomalies
- Hard to parallelize

# Count number of values

Problem

Domain of the attribute is $\{1, \dots M\}$

Piece of cake if memory available... What if the memory available is $log(M)$ ?

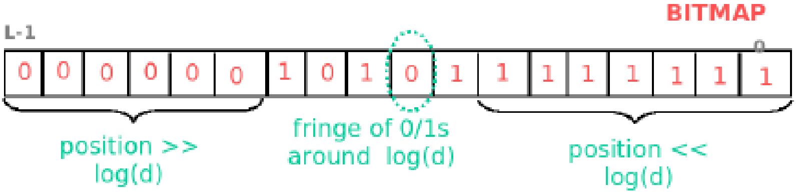
Flajolet-Martin 1983

Based on hashing: $\{1, \dots M\} \mapsto \{0, \dots 2^L\}$ with $L = log(M)$.

$$x \rightarrow \ hash(x) = y \ \rightarrow \ position\ least\ significant\ bit, lsb(x)$$

# Count number of values, followed

Init: $BITMAP(\{0, \ldots L\}) = 0$
Loop: Read $x$, $BITMAP(lsb(x)) = 1$




Result

$$R = \text{ position of rightmost 0 in } H$$

$$M \approx 2^R/.7735$$

# Decision Trees for Data Streaming

**Principle**

Grow the tree if evidence best attribute $>$ second best

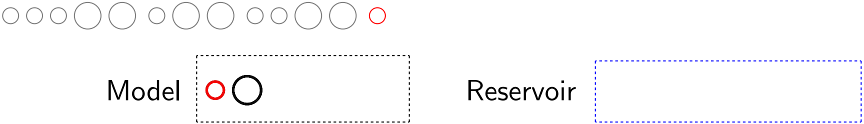
**Algorithm** parameter: confidence $\delta$ (user-defined)

While true
- Read example, propagate until a leaf
- If enough examples in leaf
  - Compute IG for all attributes;
  - $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}}$
  - Keep best if IG(best) - IG(second best ) $> \epsilon$

Mining High Speed Data Streams, Pedro Domingos, Geoffrey Hulten, KDD-00

# Stream clustering

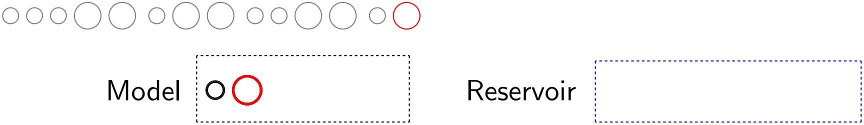
# Stream clustering




Does $e_t$ fit the current model ??

- if yes, update the model
- otherwise, put outlier $e_t$ in reservoir

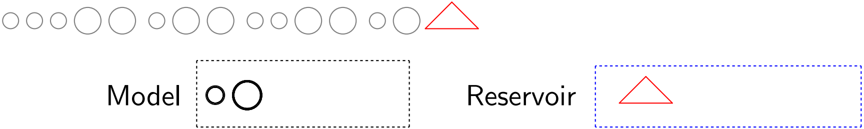
# Stream clustering




Does $e_t$ fit the current model ??

- if yes, update the model
- otherwise, put outlier $e_t$ in reservoir

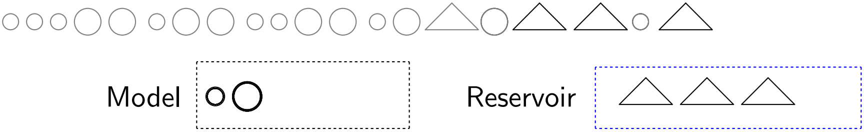
# Stream clustering




Does $e_t$ fit the current model ??

- if yes, update the model
- otherwise, put outlier $e_t$ in reservoir

# Stream clustering

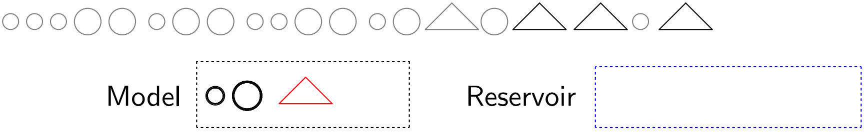



Has the distribution changed ?

- if yes, rebuild the model
- otherwise, continue

CHANGE TEST

# Stream clustering

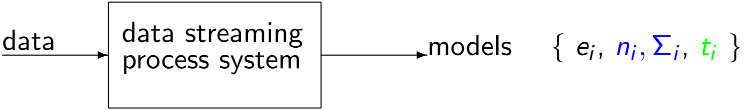



Has the distribution changed ?

CHANGE TEST

- if yes, rebuild the model
- otherwise, continue

# Strap




## Does $e_t$ fit the current model ?

- if yes, update the model
- otherwise, put $e_t$ in reservoir

## Has the distribution changed ?

- if yes, rebuild the model
- otherwise, continue

# Update the model

Stream Model: $\{(e_i, n_i, \Sigma_i, t_i)\}$

- $e_i$ examplar
- $n_i$ number of items represented by $e_i$
- $\Sigma_i$ sum of distortions incurred by $e_i$
- $t_i$ last time step when a point was affected to $e_i$

Update with decay: $\Delta$: time window

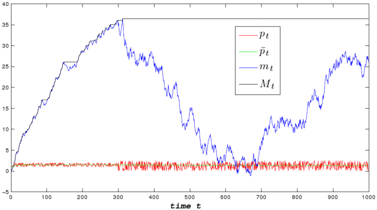
$$
\begin{aligned}
n_i &:= n_i \times \left(\frac{\Delta}{\Delta+(t-t_i)} + \frac{1}{n_i+1}\right) \\
\Sigma_i &:= \Sigma_i \times \frac{\Delta}{\Delta+(t-t_i)} + \frac{n_i}{n_i+1} d(e_t, e_i)^2 \\
t_i &:= t
\end{aligned}
$$

# Rebuild the model

Trigger

- when reservoir is full
- when changes are detected

Page-Hinkley statistic




$$\bar{p}_t = \frac{1}{t}\sum_{\ell=1}^{t} p_\ell$$
$$m_t = \sum_{\ell=1}^{t} (p_\ell - \bar{p}_\ell + \delta)$$
$$PH_t = max\{m_\ell\} - m_t$$

**HINKLEY D. Inference about the change-point in a sequence of random variables. Biometrika, 1970**

**PAGE E. Continuous inspection schemes. Biometrika, 1954**

# Experimental validation

## Data used

- Artificial dataset
- Real world data: KDD99 data
  - intrusion detection benchmark
  - 494,021 network connection records in $\mathbb{R}^{34}$
  - 23 classes: 1 normal + 22 attacks
- Baseline: DenStream

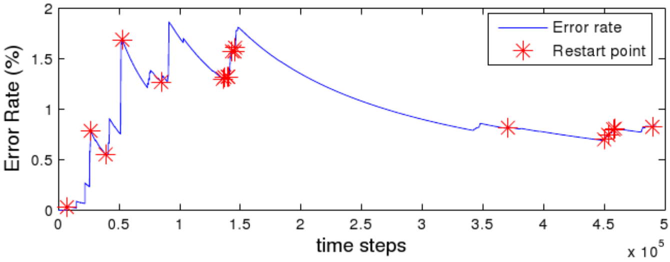
  *F. Cao, M. Ester, W. Qian, A. Zhou. Density-Based Clustering over an Evolving Data Stream with Noise. SDM 2006.*

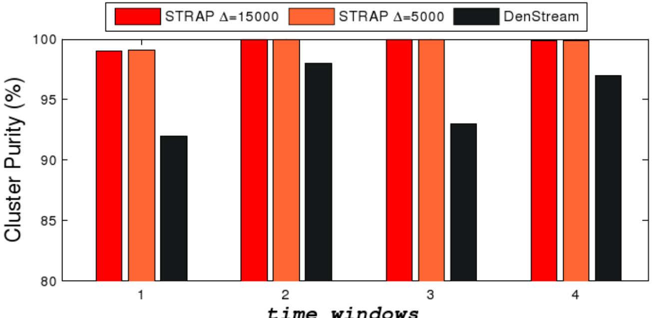
## Performance indicator

- Distortion
- Clustering accuracy / Clustering purity (supervised setting)

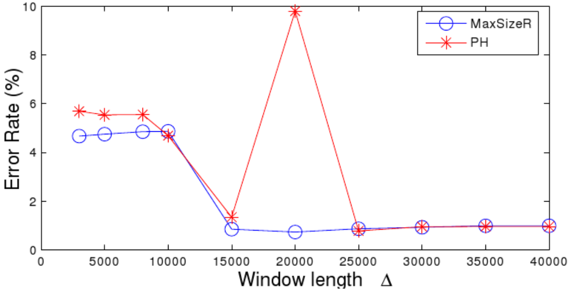
**KDD Cup 1999 data: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.**

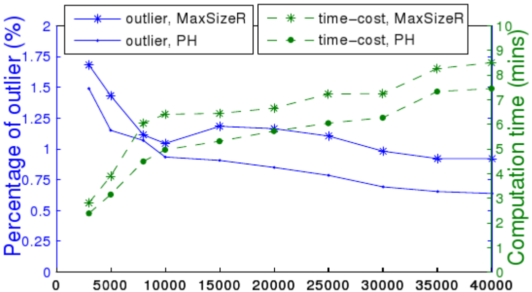
# Accuracy along time



# Restart criteria: MaxSizeR vs PH



# Discussion

Rebuild: ReservoirSize vs PH

- PH is 10% better than ReservoirSize
- PH is less stable

Strap vs DenStream

- Pros
  - better accuracy
  - model available at any time
- Cons
  - DenStream: 7 seconds
  - Strap : 7 mins

# Conclusion

Scalability: Hi-WAP

- Reduce complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N^{3/2})$
- iteratively reduce toward $\mathcal{O}(N^{(1+\gamma)})$

Stream clustering: Strap

- Hybridized with an efficient change detection method, Page-Hinkley
- Model available at any time
- BUT: slower than DenStream

Future work Provide an upper bound on the distortion loss caused by Hi-WAP

# Open issues

**What's new**

- Forget about iid;
- Forget about more than linear complexity (and log space)

**Challenges**

- Online, Anytime algs
- Distributed alg.
- Criteria of performance
- Integration of change detection