

*Cluster Ensembles - A Knowledge
Reuse Framework for Combining
Multiple Partitions*

Alexander Strehl and Joydeep Ghosh

Department of Electrical and Computer Engineering

UT Austin

Journal of Machine Learning Research 3 (2002)

Outline

- The problem of combining multiple clusterings;
- Suitable objective function for determining a single consensus clustering;
- Greedy approaches to optimize this objective function.

Notation

$X = \{x_1, \dots, x_n\}$: set of data

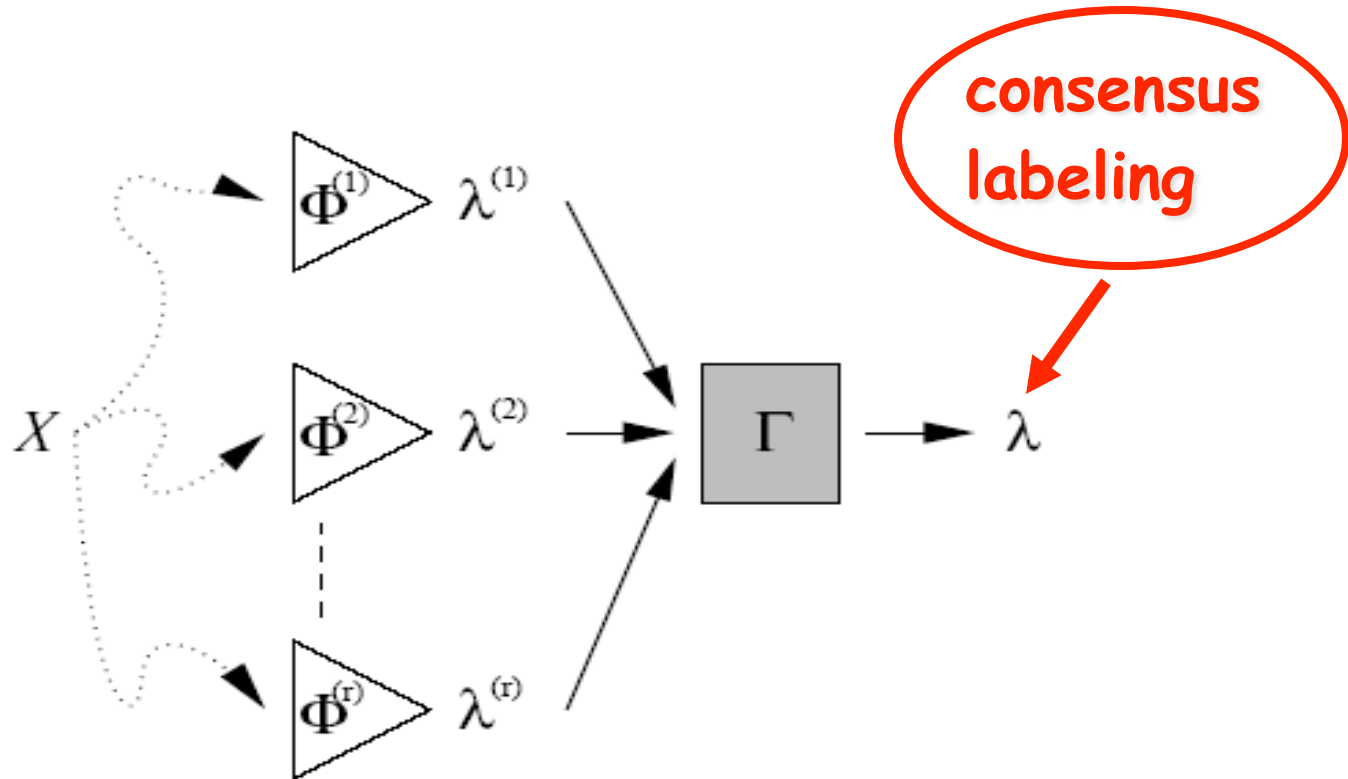
$\{C_l \mid l = 1, \dots, k\}$: partitioning of n data into k clusters

$\lambda \in \aleph^n$: label vector representing a partition
of n data into k clusters

Clusterer Φ : is a function that delivers a label vector
given a tuple of data X

$$\Phi : X \rightarrow \lambda$$

The Cluster Ensemble



A consensus function Γ combines clusterings $\lambda^{(q)}, q = 1, \dots, r$

Assumptions

The consensus function $\Gamma : \{\lambda^{(1,\dots,r)}\} \rightarrow \lambda$

has access only to the r label vectors.

The function Γ has no knowledge of :

- the original feature measurements X ;
- the clustering algorithms Φ

Example

$$\lambda^{(1)} = (1, 1, 1, 2, 2, 3, 3)^\dagger$$

$$\lambda^{(2)} = (2, 2, 2, 3, 3, 1, 1)^\dagger$$

$$\lambda^{(3)} = (1, 1, 2, 2, 3, 3, 3)^\dagger$$

$$\lambda^{(4)} = (1, 2, ?, 1, 2, ?, ?)^\dagger$$

$$n = 7; r = 4;$$

$$k^{(1)} = 3, k^{(2)} = 3, k^{(3)} = 3, k^{(4)} = 2.$$

Objective: find a “good” combined clustering with 3 clusters.

How is “good” defined?

Intuitively, a good combined clustering should *share as much information as possible* with the given 4 clusterings

Example (Contd.)

$$\lambda^{(1)} = (1, 1, 1, 2, 2, 3, 3)^\dagger$$

$$\lambda^{(2)} = (2, 2, 2, 3, 3, 1, 1)^\dagger$$

$$\lambda^{(3)} = (1, 1, 2, 2, 3, 3, 3)^\dagger$$

$$\lambda^{(4)} = (1, 2, ?, 1, 2, ?, ?)^\dagger$$

$$n = 7; r = 4;$$

$$k^{(1)} = 3, k^{(2)} = 3, k^{(3)} = 3, k^{(4)} = 2.$$

Objective: find a "good" combined clustering with 3 clusters.

A good clustering choice seems

$$\lambda = (1, 1, 1, 2, 2, 3, 3)$$

In fact, it can be shown that this clustering shares the maximum information with the given four label vectors.

Assume canonical forms of label vectors.

Objective Function for Cluster Ensembles

$$\Gamma : \{ \lambda^{(q)} \mid q \in \{1, \dots, r\} \} \rightarrow \lambda$$

$$\Lambda = \{ \lambda^{(q)} \mid q \in \{1, \dots, r\} \}$$

In absence of a-priori information, a reasonable goal is to seek a clustering that shares the most information with the original clusterings.

How can we measure the "shared information" between clusterings?

Use Mutual Information.

Mutual Information

- Symmetric measure to quantify the statistical information shared between two distributions.
- It is defined in terms of Entropy H .

Let X and Y be random variables.

$H(X)$: measure the uncertainty in the random variable X (entropy)

$$I(X,Y) = H(X) - H(X|Y)$$
$$= \sum_x \sum_y \Pr(x,y) \log \frac{\Pr(x,y)}{\Pr(x)\Pr(y)}$$

Describes the information on X provided by Y

Mutual Information

➤ So: The mutual information provides a sound measure to quantify the information shared between pair of clusterings.

X : random variable described by the cluster labeling $\lambda^{(a)}$

Y : random variable described by the cluster labeling $\lambda^{(b)}$

$I(X,Y)$ = shared information between $\lambda^{(a)}$ and $\lambda^{(b)}$

It can be shown that $I(X,Y)$ is a metric (with no upper bound)

For easier interpretation :

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}}$$

Since $I(X,X) = H(X)$, $NMI(X,X) = 1$.

Estimation of NMI

$NMI(X,Y)$ needs to be estimated using the sampled quantities provided by the clusterings.

$k^{(a)}, k^{(b)}$: number of clusters in $\lambda^{(a)}$ and $\lambda^{(b)}$ respectively

$n_h^{(a)}$: number of data in cluster C_h according to $\lambda^{(a)}$

$n_l^{(b)}$: number of data in cluster C_l according to $\lambda^{(b)}$

$n_{h,l}$: number of data in both clusters C_h and C_l

$$\phi^{(\text{NMI})}(\lambda^{(a)}, \lambda^{(b)}) = \frac{\sum_{h=1}^{k^{(a)}} \sum_{\ell=1}^{k^{(b)}} n_{h,\ell} \log \left(\frac{n \cdot n_{h,\ell}}{n_h^{(a)} n_\ell^{(b)}} \right)}{\sqrt{\left(\sum_{h=1}^{k^{(a)}} n_h^{(a)} \log \frac{n_h^{(a)}}{n} \right) \left(\sum_{\ell=1}^{k^{(b)}} n_\ell^{(b)} \log \frac{n_\ell^{(b)}}{n} \right)}}.$$

Average Normalized Mutual Information (ANMI)

Recall: $\Lambda = \{\lambda^{(q)} \mid q \in \{1, \dots, r\}\}$

We can now define a measure between a set of r labelings Λ and a single labeling $\hat{\lambda}$

$$\phi^{(\text{ANMI})}(\Lambda, \hat{\lambda}) = \frac{1}{r} \sum_{q=1}^r \phi^{(\text{NMI})}(\hat{\lambda}, \lambda^{(q)}).$$

Optimal Combined Clustering

The optimal combined clustering $\lambda^{(k-opt)}$ is the one that has the maximal ANMI with all individual labelings in Λ , given that the number of consensus clusters is k .

$$\lambda^{(k-opt)} = \arg \max_{\hat{\lambda}} \sum_{q=1}^r \phi^{(NMI)}(\hat{\lambda}, \lambda^{(q)})$$

Greedy Optimization Scheme

1. Initial labeling: single labeling that gives highest ANMI;
2. For each data, its label is changed to each of the other possible $k-1$ labels, and the ANMI objective is re-evaluated;
3. If ANMI increases, the data's label is changed to the best new value, and the algorithm proceeds to the next point;
4. When all data have been checked for possible improvements, an iteration is complete;
5. If at least one label was changed during the last iteration, go back to 2., otherwise terminate (local minimum is reached).

Expensive, dependency on initial labeling, poor local minima.

More Efficient Greedy Approaches

- ❖ Intuitive heuristics;
- ❖ Do not perform a direct maximization of ANMI objective;
- ❖ Three different algorithms:
 - Cluster-based Similarity Partitioning Algorithm (CSPA)
 - HyperGraph Partitioning Algorithm (HGPA)
 - Meta-CLustering Algorithm (MCLA)
- ❖ All three algorithms first transform the set of clustering into a hypergraph representation.

Mapping a set of Clusterings to a Hypergraph

- ❖ Hypergraph: vertices and hyperedges;
- ❖ Hyperedge: generalization of an edge; it can connect any set of vertices;

For each label vector $\lambda^{(q)} \in \mathfrak{X}^n$, a binary membership indicator matrix $\mathbf{H}^{(q)}$ is constructed, with a column for each cluster (corresponding to a hyperedge).

Hypergraph: Example

	$\lambda^{(1)}$	$\lambda^{(2)}$	$\lambda^{(3)}$	$\lambda^{(4)}$		$H^{(1)}$			$H^{(2)}$			$H^{(3)}$			$H^{(4)}$		
						h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}	
x_1	1	2	1	1		v_1	1	0	0	0	1	0	1	0	0	1	0
x_2	1	2	1	2		v_2	1	0	0	0	1	0	1	0	0	0	1
x_3	1	2	2	?	\Leftrightarrow	v_3	1	0	0	0	1	0	0	1	0	0	0
x_4	2	3	2	1		v_4	0	1	0	0	0	1	0	1	0	1	0
x_5	2	3	3	2		v_5	0	1	0	0	0	1	0	0	1	0	1
x_6	3	1	3	?		v_6	0	0	1	1	0	0	0	0	1	0	0
x_7	3	1	3	?		v_7	0	0	1	1	0	0	0	0	1	0	0

original
label
vectors

The concatenated block matrix

$$H = H^{(1,\dots,4)} = \left(H^{(1)} \dots H^{(4)} \right)$$

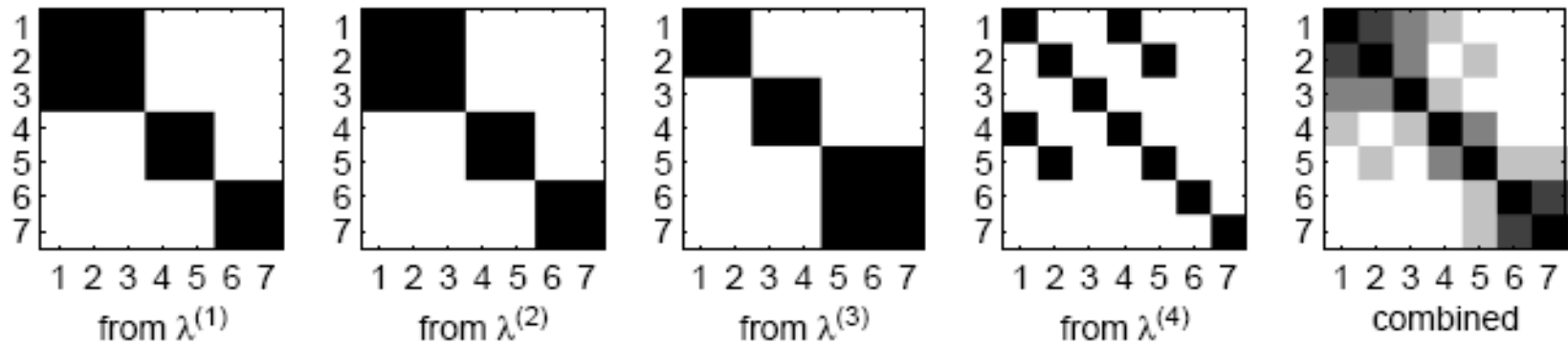
defines the adjacency matrix of a hypergraph
with 7 vertices and 11 hyperedges

Cluster-based Similarity Partitioning Algorithm (CSPA)

- ❖ Two points have similarity 1 if they belong to the same cluster; similarity 0 otherwise.
- ❖ This defines a binary similarity matrix for each clustering.
- ❖ Overall similarity matrix S : entry-wise average of the r above matrices.
- ❖ The element S_{ij} of S represents the fraction of clusterings in which two data x_i and x_j are in the same cluster.

$$S = \frac{1}{r} HH^T$$

Cluster-based Similarity Partitioning Algorithm (CSPA) -- Example



	$\lambda^{(1)}$	$\lambda^{(2)}$	$\lambda^{(3)}$	$\lambda^{(4)}$		$H^{(1)}$			$H^{(2)}$			$H^{(3)}$			$H^{(4)}$	
						h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}
x_1	1	2	1	1	v_1	1	0	0	0	1	0	1	0	0	1	0
x_2	1	2	1	2	v_2	1	0	0	0	1	0	1	0	0	0	1
x_3	1	2	2	?	$\Leftrightarrow v_3$	1	0	0	0	1	0	0	1	0	0	0
x_4	2	3	2	1	v_4	0	1	0	0	0	1	0	1	0	1	0
x_5	2	3	3	2	v_5	0	1	0	0	0	1	0	0	1	0	1
x_6	3	1	3	?	v_6	0	0	1	1	0	0	0	0	1	0	0
x_7	3	1	3	?	v_7	0	0	1	1	0	0	0	0	1	0	0

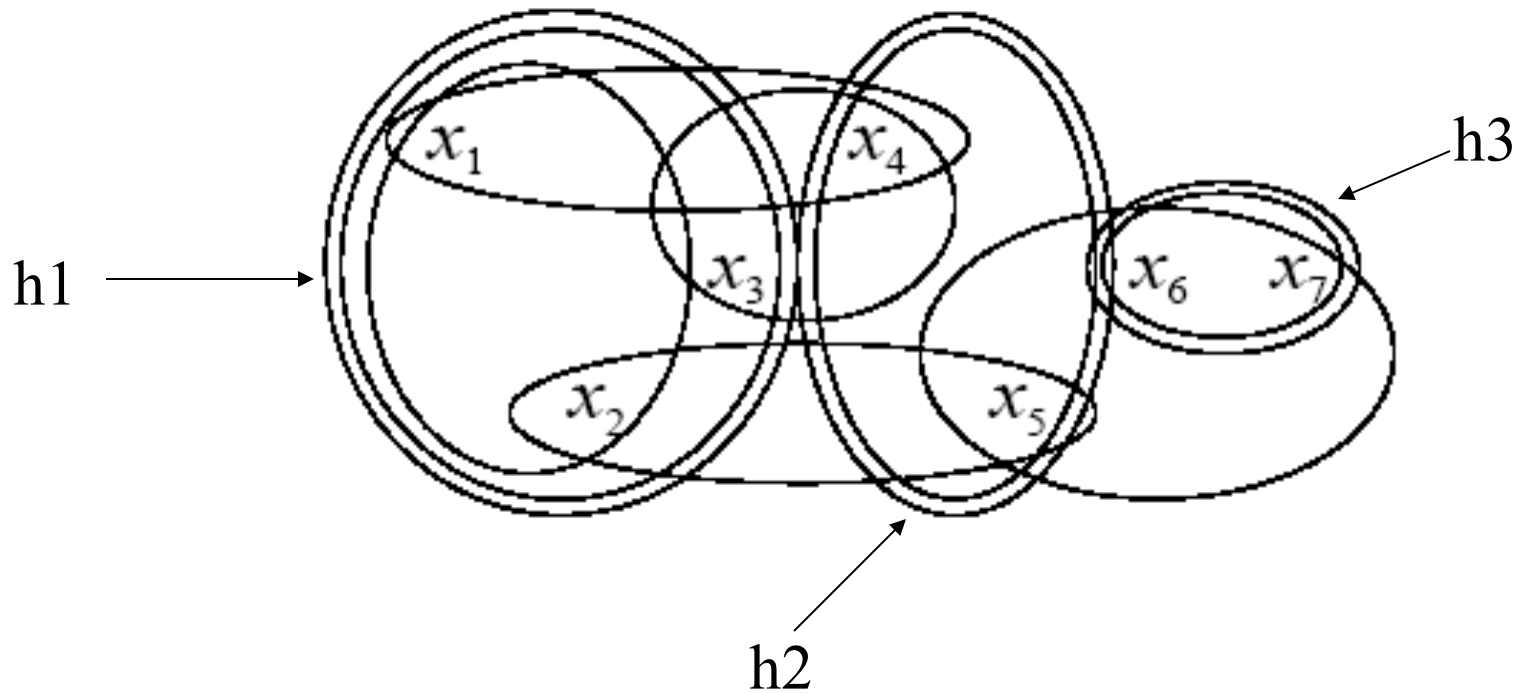
Cluster-based Similarity Partitioning Algorithm (CSPA)

- ❖ The combined similarity matrix is used to recluster the data using a similarity based clustering algorithm.
- ❖ A graph-partitioning based clustering approach is used:
 - ❖ The induced similarity graph (vertex = data; edge weight = similarity) is partitioned using METIS.
- ❖ Advantages:
 - ❖ Simple heuristic.
- ❖ Disadvantages:
 - ❖ Computational and storage complexity are both quadratic in n .

HyperGraph-Partitioning Algorithm (HGPA)

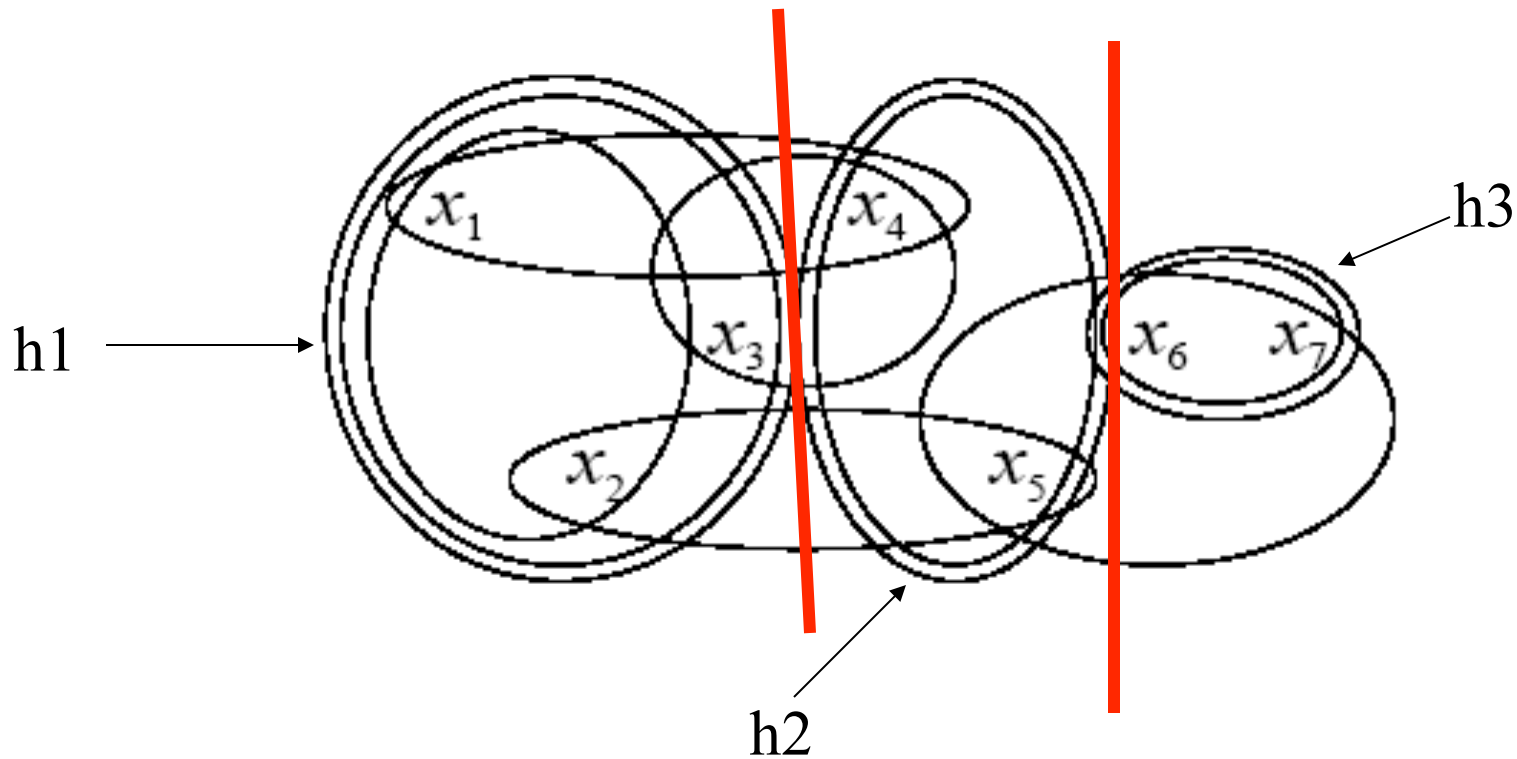
- ❖ Seeks a partitioning of the hypergraph by cutting a minimal number of hyperedges.
- ❖ All hyperedges have the same weight.
- ❖ This algorithm looks for a hyperedge separator that partitions the hypergraph into k unconnected components of *approximately the same size*.
- ❖ Captures n -way relationships between data points.

HyperGraph-Partitioning Algorithm (HGPA) -- Example



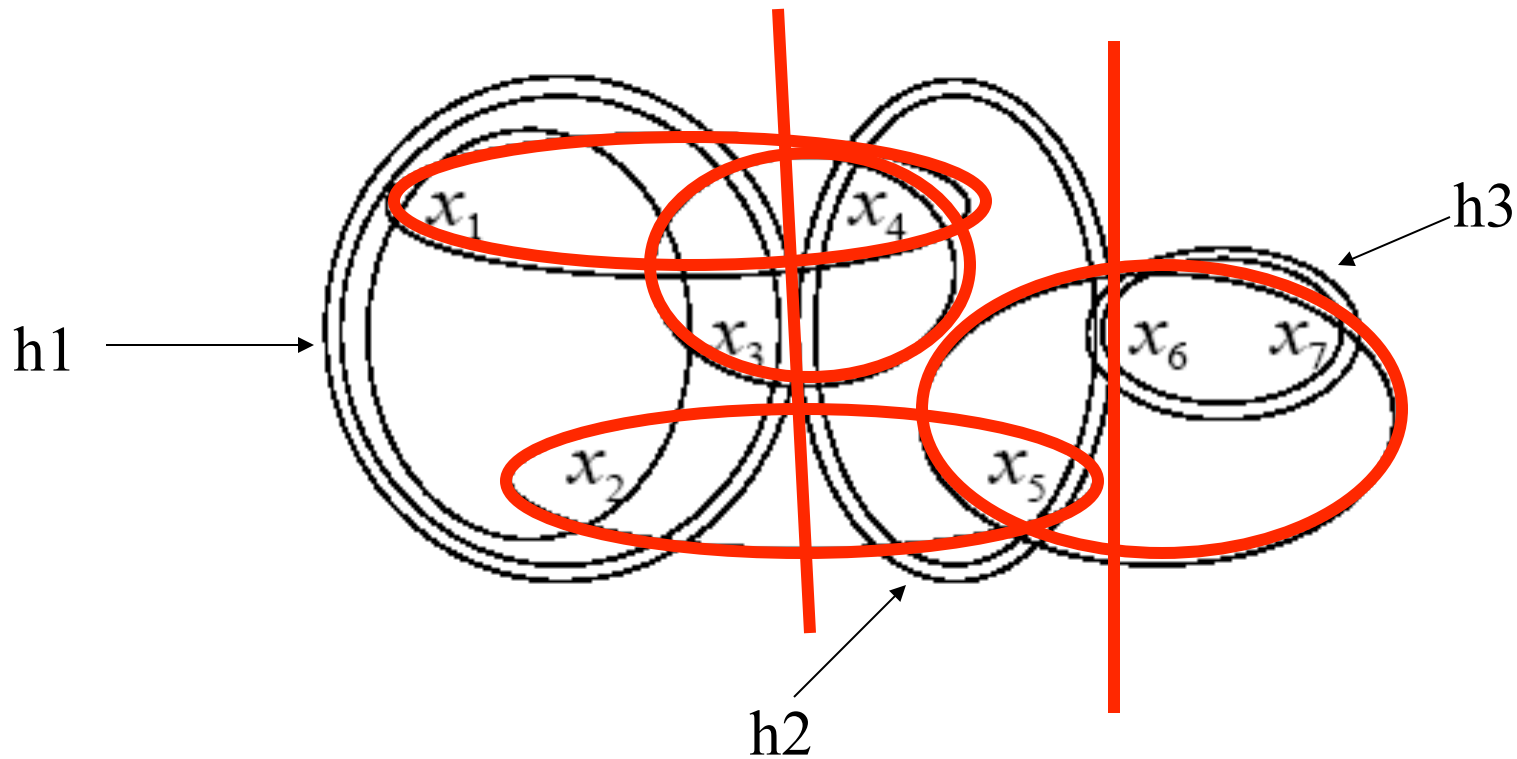
We seek a minimal hyperedge cut that partitions the seven data into 3 balanced groups.

HyperGraph-Partitioning Algorithm (HGPA) -- Example



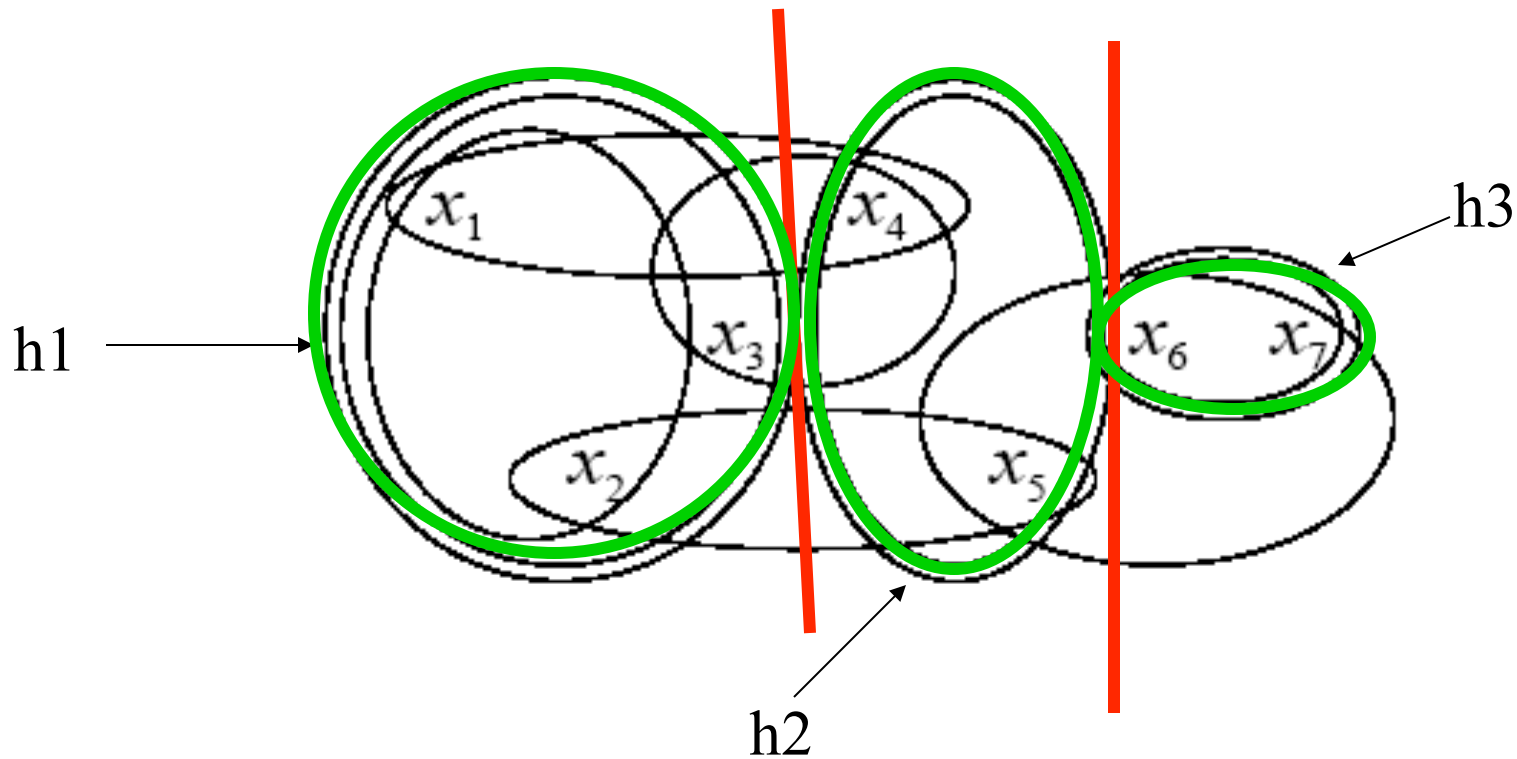
The minimal hyperedge cut is 4.

HyperGraph-Partitioning Algorithm (HGPA) -- Example



The minimal hyperedge cut is 4.

HyperGraph-Partitioning Algorithm (HGPA) -- Example



The minimal hyperedge cut is 4.

The resulting combined clustering is:

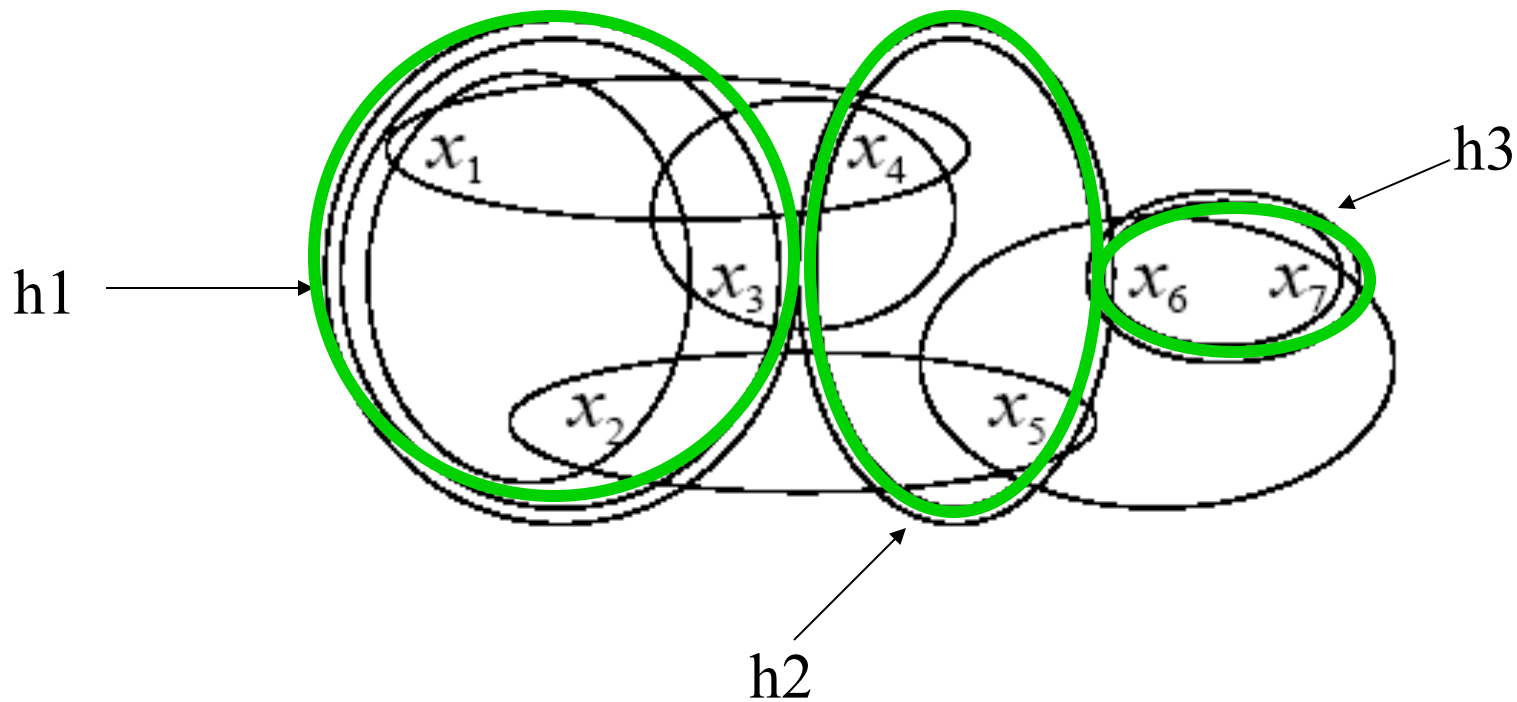
$$\{\{x_1, x_2, x_3\}, \{x_4, x_5\}, \{x_6, x_7\}\}$$

HyperGraph-Partitioning Algorithm (HGPA)

Limitation: No knowledge of partially cut hyperedges.

I.e., The algorithm does not keep track of how much of a hyperedge is left in the same group after the cut.

HyperGraph-Partitioning Algorithm (HGPA) -- Example



$$\{\{x_1, x_2, x_7\}, \{x_3, x_4\}, \{x_5, x_6\}\}, \{\{x_1, x_7\}, \{x_3, x_4\}, \{x_2, x_5, x_6\}\}$$

Both partitionings cut all three hyperedges.

HyperGraph-Partitioning Algorithm (HGPA)

Advantage: Efficiency. Complexity is $O(nkr)$
(assuming usage of HMETIS).

Meta-Clustering Algorithm (MCLA)

- ❖ Based on clustering of clusters.
- ❖ Provides object-wise confidence estimates of cluster membership.
- ❖ Basic idea:
 - ❖ Each cluster corresponds to a hyperedge;
 - ❖ Hyperedges are grouped, and each data is assigned to the collapsed hyperedge to which it participates more strongly;

Initial number of hyperedges: $\sum_{q=1}^r k^{(q)}$

After collapsing hyperedges: k

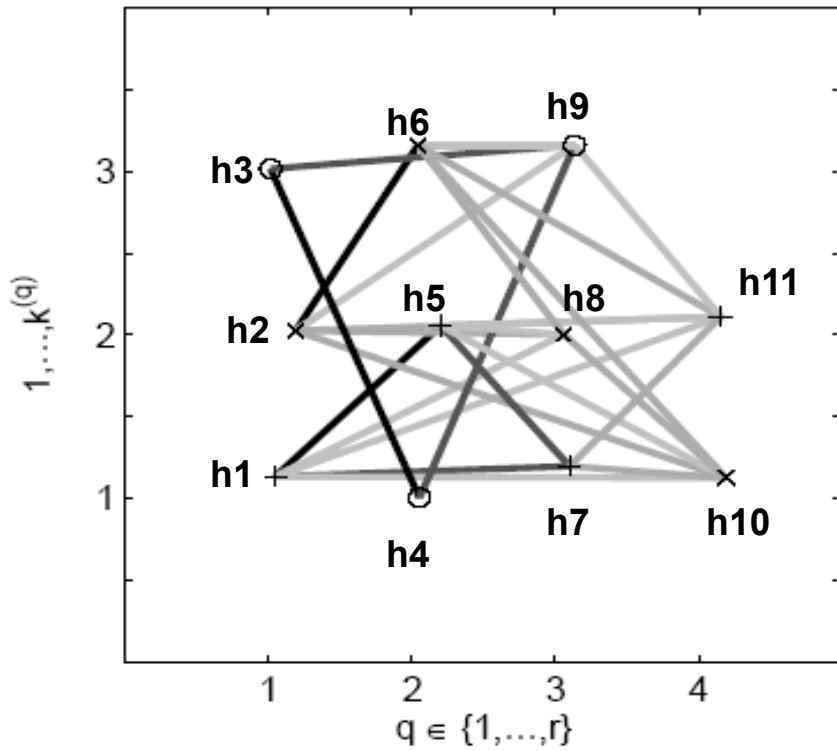
Meta-Clustering Algorithm (MCLA) - The Steps

- ❖ Construct Meta-graph: View the indicator vectors h (hyperedges of H) as vertices of another regular undirected graph (the meta-graph). The edges have weights proportional to the *similarity* between vertices.
- ❖ Binary Jaccard measure:

$$w_{a,b} = \frac{\mathbf{h}_a^\dagger \mathbf{h}_b}{\|\mathbf{h}_a\|_2^2 + \|\mathbf{h}_b\|_2^2 - \mathbf{h}_a^\dagger \mathbf{h}_b}$$

Corresponds to the ratio of the cardinality of the intersection to the union of the sets of data of the two hyperedges.

Construction of Meta-graph - Example

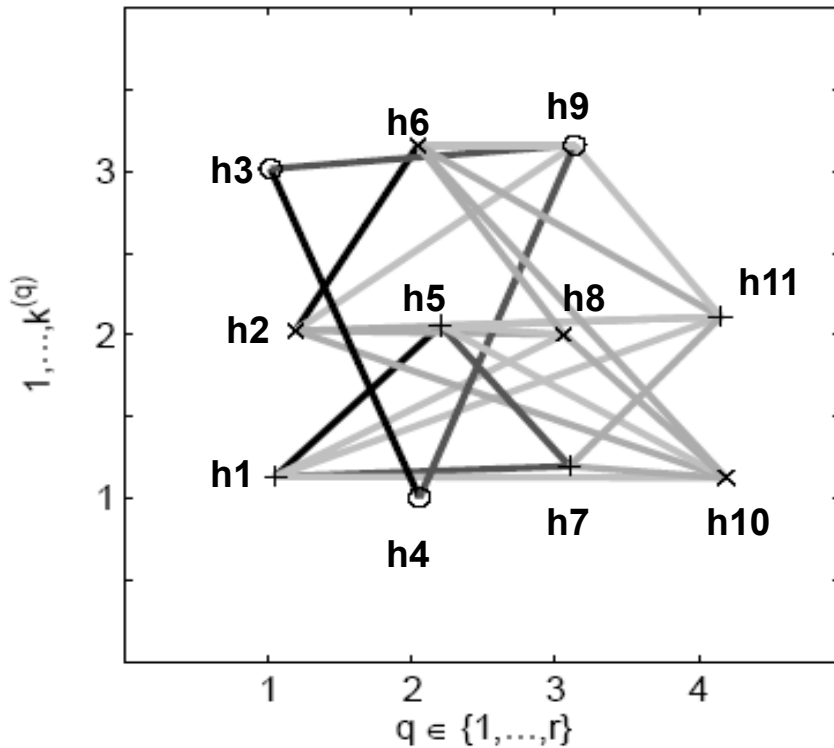


	$H^{(1)}$			$H^{(2)}$			$H^{(3)}$			$H^{(4)}$	
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}
v_1	1	0	0	0	1	0	1	0	0	1	0
v_2	1	0	0	0	1	0	1	0	0	0	1
v_3	1	0	0	0	1	0	0	1	0	0	0
v_4	0	1	0	0	0	1	0	1	0	1	0
v_5	0	1	0	0	0	1	0	0	1	0	1
v_6	0	0	1	1	0	0	0	0	1	0	0
v_7	0	0	1	1	0	0	0	0	1	0	0

Meta-Clustering Algorithm (MCLA) - The Steps

- ❖ Cluster Hyperedges: partitions the meta-graph into k balanced meta-clusters.
- ❖ This results in a clustering of the h vectors.

Clustering Hyperedges - Example



The resulting meta - clusters are :

$$C_1^{(M)} = \{h_3, h_4, h_9\}$$

$$C_2^{(M)} = \{h_2, h_6, h_8, h_{10}\}$$

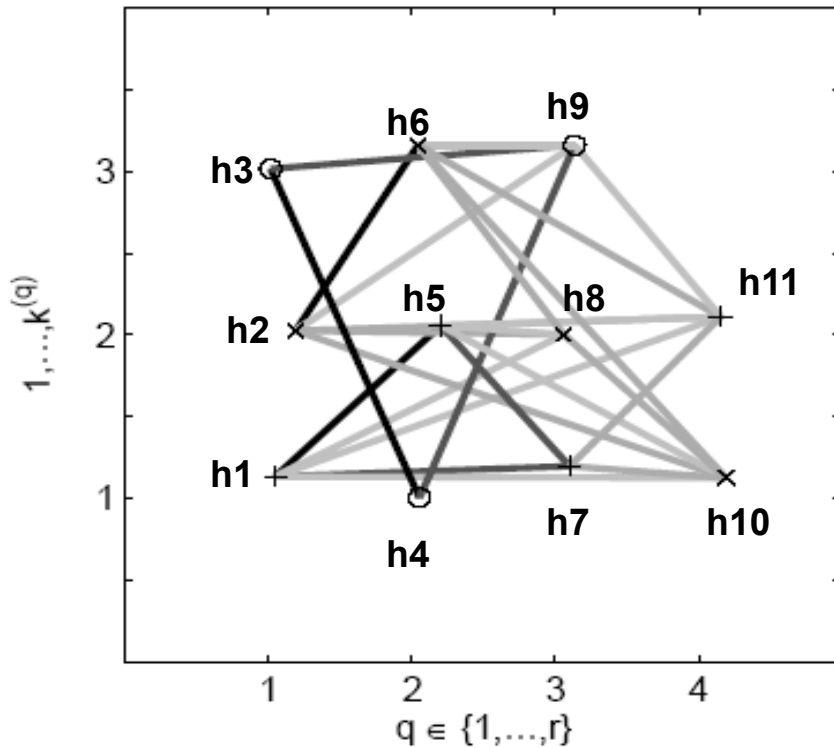
$$C_3^{(M)} = \{h_1, h_5, h_7, h_{11}\}$$

	$H^{(1)}$			$H^{(2)}$			$H^{(3)}$			$H^{(4)}$	
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}
v_1	1	0	0	0	1	0	1	0	0	1	0
v_2	1	0	0	0	1	0	1	0	0	0	1
v_3	1	0	0	0	1	0	0	1	0	0	0
v_4	0	1	0	0	0	1	0	1	0	1	0
v_5	0	1	0	0	0	1	0	0	1	0	1
v_6	0	0	1	1	0	0	0	0	1	0	0
v_7	0	0	1	1	0	0	0	0	1	0	0

Meta-Clustering Algorithm (MCLA) - The Steps

- ❖ Collapse Meta-clusters: the hyperedges of each meta-cluster are collapsed into a single meta-hyperedge.
- ❖ This is achieved by averaging all the h vectors of a particular meta-cluster .
- ❖ The entries of the meta-hyperedge indicate the level of association of objects with the corresponding meta-cluster.

Collapsing Meta-clusters - Example



The resulting meta - clusters are :

$$C_1^{(M)} = \{h_3, h_4, h_9\}$$

$$C_2^{(M)} = \{h_2, h_6, h_8, h_{10}\}$$

$$C_3^{(M)} = \{h_1, h_5, h_7, h_{11}\}$$

	$H^{(1)}$			$H^{(2)}$			$H^{(3)}$			$H^{(4)}$	
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}
v_1	1	0	0	0	1	0	1	0	0	1	0
v_2	1	0	0	0	1	0	1	0	0	0	1
v_3	1	0	0	0	1	0	0	1	0	0	0
v_4	0	1	0	0	0	1	0	1	0	1	0
v_5	0	1	0	0	0	1	0	0	1	0	1
v_6	0	0	1	1	0	0	0	0	1	0	0
v_7	0	0	1	1	0	0	0	0	1	0	0

Collapsing Meta-clusters - Example

$$C_1^{(M)} = \{h_3, h_4, h_9\} \quad \Rightarrow (0, 0, 0, 0, 1/3, 1, 1)$$

$$C_2^{(M)} = \{h_2, h_6, h_8, h_{10}\}$$

$$C_3^{(M)} = \{h_1, h_5, h_7, h_{11}\}$$

	H ⁽¹⁾		H ⁽²⁾		H ⁽³⁾		H ⁽⁴⁾				
	h ₁	h ₂	h ₃	h ₄	h ₅	h ₆	h ₇	h ₈	h ₉	h ₁₀	h ₁₁
v ₁	1	0	0	0	1	0	1	0	0	1	0
v ₂	1	0	0	0	1	0	1	0	0	0	1
v ₃	1	0	0	0	1	0	0	1	0	0	0
v ₄	0	1	0	0	0	1	0	1	0	1	0
v ₅	0	1	0	0	0	1	0	0	1	0	1
v ₆	0	0	1	1	0	0	0	0	1	0	0
v ₇	0	0	1	1	0	0	0	0	1	0	0

Collapsing Meta-clusters - Example

$$C_1^{(M)} = \{h_3, h_4, h_9\} \Rightarrow (0, 0, 0, 0, 1/3, 1, 1)$$

$$C_2^{(M)} = \{h_2, h_6, h_8, h_{10}\} \Rightarrow (1/4, 0, 1/4, 1, 1/2, 0, 0)$$

$$C_3^{(M)} = \{h_1, h_5, h_7, h_{11}\}$$

	H ⁽¹⁾			H ⁽²⁾			H ⁽³⁾			H ⁽⁴⁾	
	h ₁	h ₂	h ₃	h ₄	h ₅	h ₆	h ₇	h ₈	h ₉	h ₁₀	h ₁₁
v ₁	1	0	0	0	1	0	1	0	0	1	0
v ₂	1	0	0	0	1	0	1	0	0	0	1
v ₃	1	0	0	0	1	0	0	1	0	0	0
v ₄	0	1	0	0	0	1	0	1	0	1	0
v ₅	0	1	0	0	0	1	0	0	1	0	1
v ₆	0	0	1	1	0	0	0	0	1	0	0
v ₇	0	0	1	1	0	0	0	0	1	0	0

Collapsing Meta-clusters - Example

$$C_1^{(M)} = \{h_3, h_4, h_9\} \Rightarrow (0, 0, 0, 0, 1/3, 1, 1)$$

$$C_2^{(M)} = \{h_2, h_6, h_8, h_{10}\} \Rightarrow (1/4, 0, 1/4, 1, 1/2, 0, 0)$$

$$C_3^{(M)} = \{h_1, h_5, h_7, h_{11}\} \Rightarrow (3/4, 1, 1/2, 0, 1/4, 0, 0)$$

	H ⁽¹⁾			H ⁽²⁾			H ⁽³⁾			H ⁽⁴⁾	
	h ₁	h ₂	h ₃	h ₄	h ₅	h ₆	h ₇	h ₈	h ₉	h ₁₀	h ₁₁
v ₁	1	0	0	0	1	0	1	0	0	1	0
v ₂	1	0	0	0	1	0	1	0	0	0	1
v ₃	1	0	0	0	1	0	0	1	0	0	0
v ₄	0	1	0	0	0	1	0	1	0	1	0
v ₅	0	1	0	0	0	1	0	0	1	0	1
v ₆	0	0	1	1	0	0	0	0	1	0	0
v ₇	0	0	1	1	0	0	0	0	1	0	0

Meta-CLustering Algorithm (MCLA) - The Steps

- ❖ Compete for Objects: each object is assigned to the meta-cluster with the strongest association (i.e., highest entry in the association vector).

Competing for Objects - Example

$$C_1^{(M)} = \{h_3, h_4, h_9\} \quad \Rightarrow (0, 0, 0, 0, 1/3, 1, 1)$$

$$C_2^{(M)} = \{h_2, h_6, h_8, h_{10}\} \quad \Rightarrow (1/4, 0, 1/4, 1, 1/2, 0, 0)$$

$$C_3^{(M)} = \{h_1, h_5, h_7, h_{11}\} \quad \Rightarrow (3/4, 1, 1/2, 0, 1/4, 0, 0)$$

↓

$$C_1 = \{x_6, x_7\}$$

$$C_2 = \{x_4, x_5\}$$

$$C_3 = \{x_1, x_2, x_3\}$$

Meta-Clustering Algorithm (MCLA)

❖ Complexity: $O(nk^2r^2)$

Empirical Evaluation

Controlled Experiment

- ❖ $n=400$; $k=10$; clusters are balanced;
- ❖ A partition is generated at random by a random permutation;
- ❖ The partition is duplicated $r=8$ times;
- ❖ In each of the 8 labelings, a fraction of labels is replaced with random labels from a uniform distribution from 1 to $k=10$;
- ❖ The resulting noisy labelings are feeded to the proposed algorithms.

Controlled Experiment: Evaluation

Measures used are :

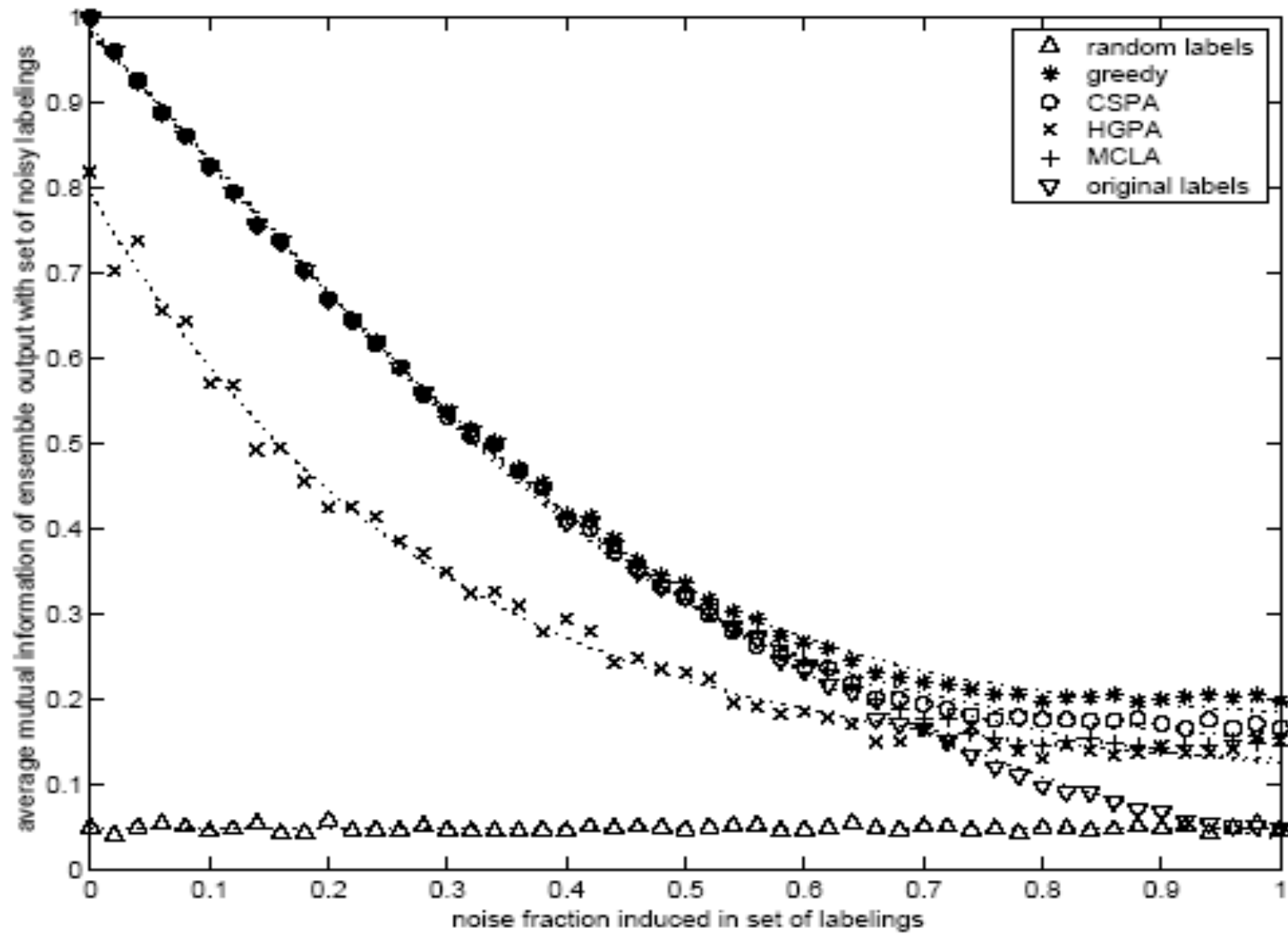
Normalized objective function $\phi^{(ANMI)}(\Lambda, \lambda)$

of the ensemble output λ with all the individual labels in Λ ;

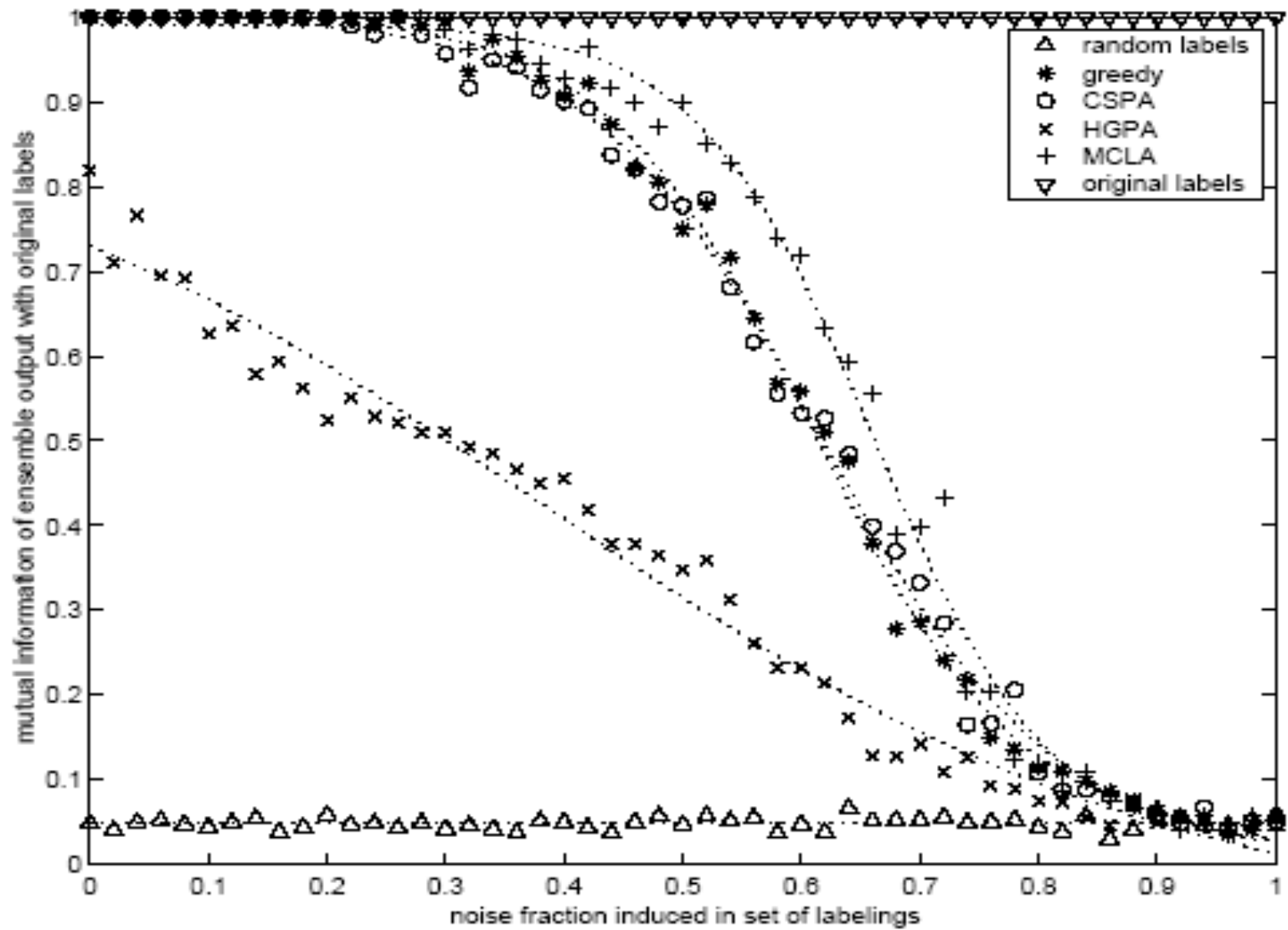
Normalized mutual information of the ensemble output with the original undistorted labeling

using $\phi^{(NMI)}(\kappa, \lambda)$

$$\phi^{(ANMI)}(\Lambda, \lambda)$$



$$\phi^{(NMI)}(\kappa, \lambda)$$



Remarks

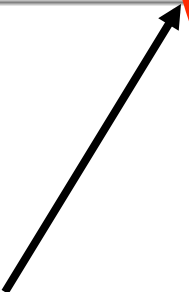
Objective function $\phi^{(ANMI)}(\Lambda, \lambda)$ seems to be a suitable choice to evaluate the algorithms in practice when $\phi^{(NMI)}(\kappa, \lambda)$ may not be available.

Findings: MCLA tends to be best in low noise/diversity scenarios; HGPA/CSPA tend to be better in high noise/diversity settings. Sounds reasonable since MCLA assumes meaningful cluster correspondences.

Data Sets

name	features	#features	#categories	balance	similarity	#clusters
2D2K	real	2	2	1.00	Euclidean	2
8D5K	real	8	5	1.00	Euclidean	5
PENDIG	real	16	10	0.87	Euclidean	10
YAH00	ordinal	2903	20	0.24	Cosine	40

**Input
parameter**



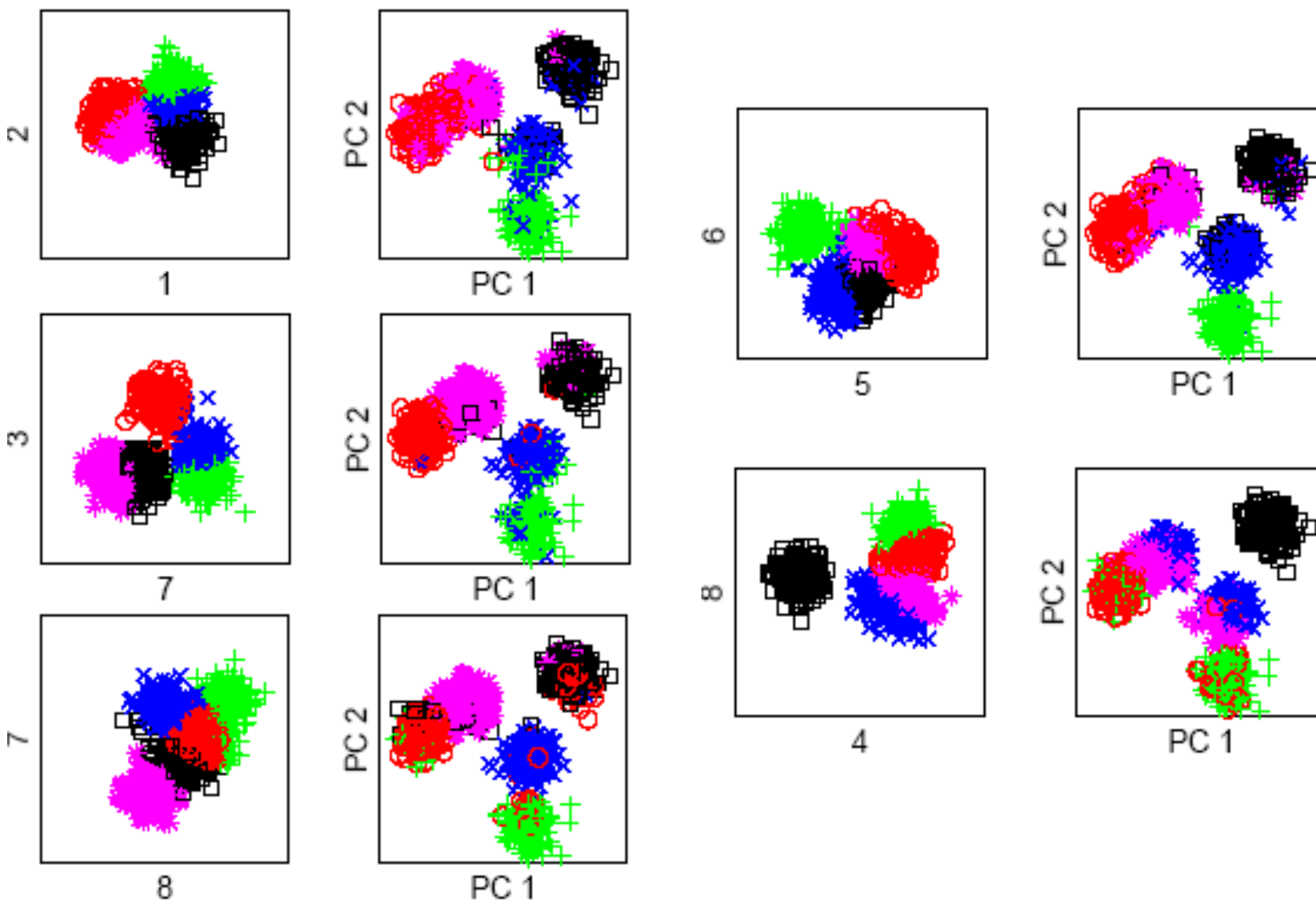
Three Scenarios

- ❖ Feature-Distributed Clustering (FDC):
 - ❖ Each entity has access only to a subset of features;
- ❖ Object-Distributed Clustering (ODC)
 - ❖ Each entity has access only to a subset of objects;
- ❖ Robust Centralized Clustering (RCC):
 - ❖ The robustness of clustering can be increased through combining a set of clusterings.

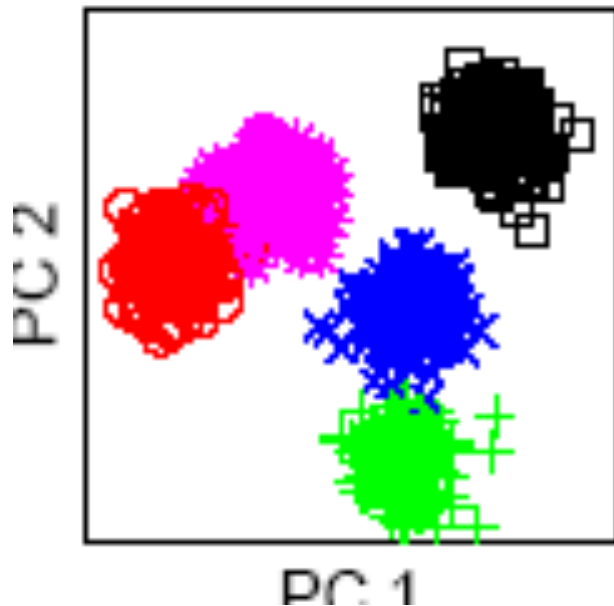
Feature-Distributed Clustering (FDC)

- ❖ Each clusterer has access to all data;
- ❖ Each clusterer has a partial view of the data (i.e., access to a subset of features);
- ❖ Each clusterer uses the same algorithm to cluster the data;
- ❖ In the combining phase, individual cluster labels are integrated.

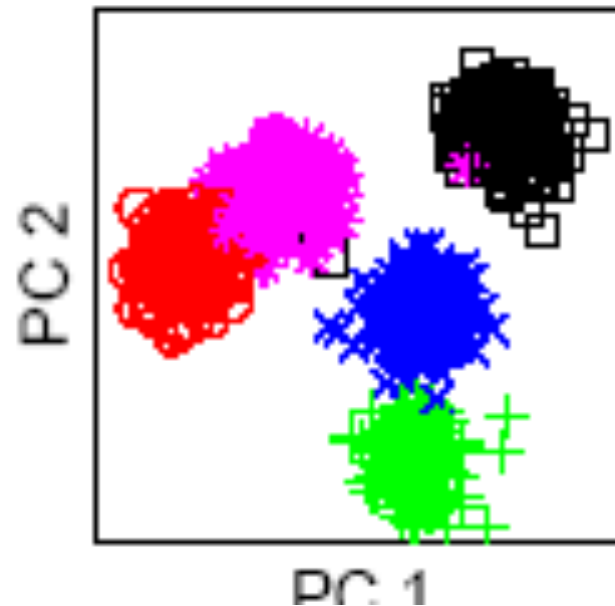
Experiments with 8D5K data



Experiments with 8D5K data



Reference clustering



Consensus clustering

Best individual result: 120 mislabeled points;
Consensus clustering: 3 mislabeled points.

FDC Results

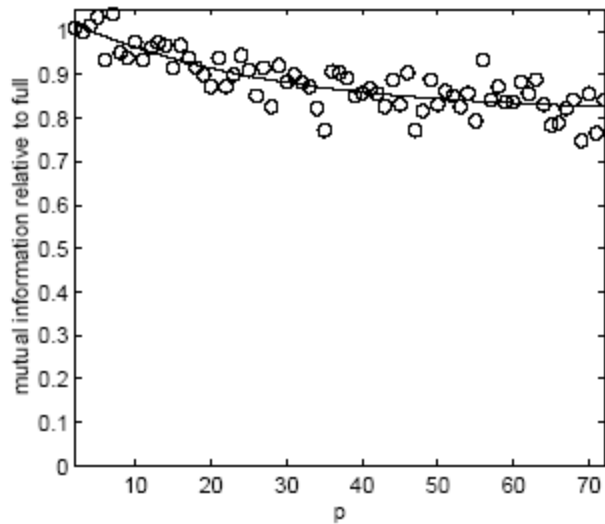
input and parameters			quality				
data	sub-space	#	all features	consensus	max subspace	average subspace	min subspace
	#dims	models	$\phi^{(\text{NMI})}(\kappa, \lambda^{(\text{all})})$	$\phi^{(\text{NMI})}(\kappa, \lambda)$	\max_q	avg_q	\min_q
		r			$\phi^{(\text{NMI})}(\kappa, \lambda^{(q)})$	$\phi^{(\text{NMI})}(\kappa, \lambda^{(q)})$	$\phi^{(\text{NMI})}(\kappa, \lambda^{(q)})$
2D2K	1	3	0.84747	0.68864	0.68864	0.64145	0.54706
8D5K	2	5	1.00000	0.98913	0.76615	0.69823	0.62134
PENDIG	4	10	0.67805	0.63918	0.47865	0.41951	0.32641
YAHOO	128	20	0.48877	0.41008	0.20183	0.16033	0.11143

The consensus clustering is as good as or better than the best individual input clustering, and always better than the average quality of individual clusterings.

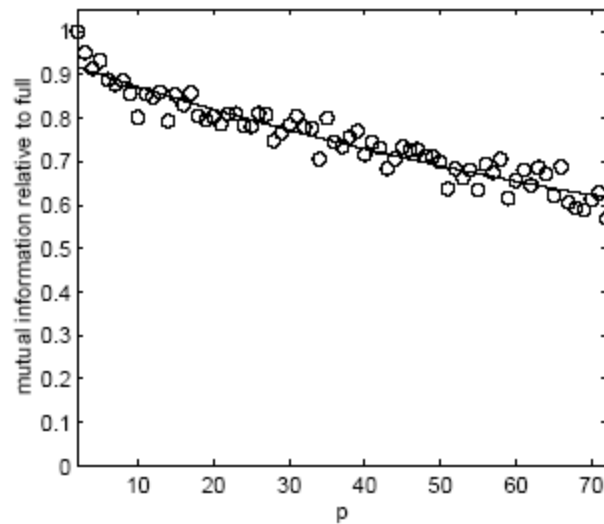
Object-Distributed Clustering (ODC)

- ❖ Each clusterer has access only to a subset of objects; thus labelings are partial;
- ❖ Sampling strategy: Objects are partitioned among the clusterers to provide full coverage; Partitions overlap.
- ❖ Each clusterer uses the same algorithm to cluster the data (into k groups)

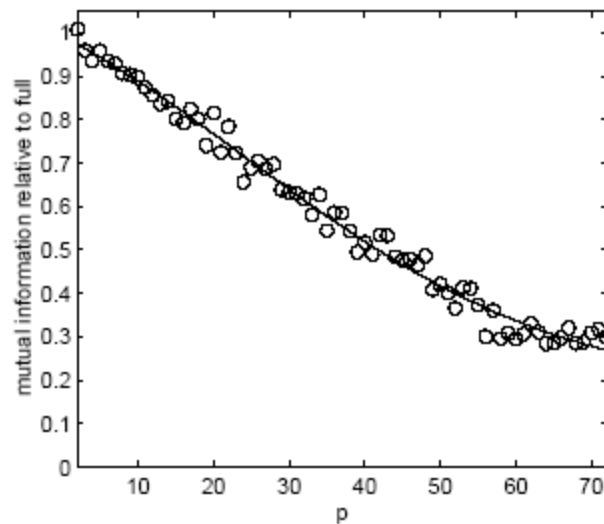
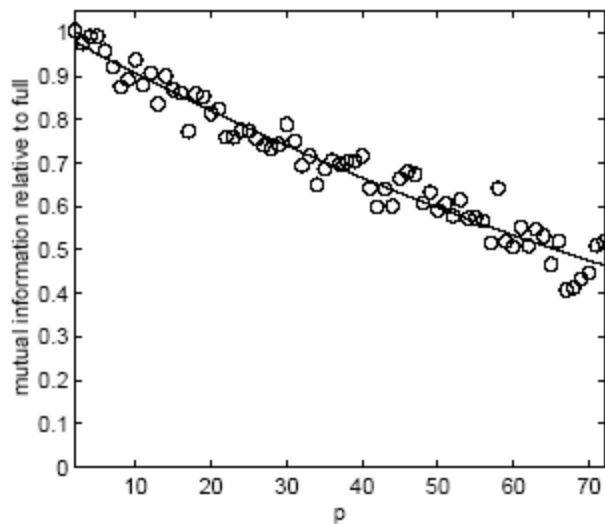
ODC Results



(a)



(b)



Robust Centralized Clustering (RCC)

- ❖ Each clusterer has access to all data and to all features;
- ❖ Each clusterer uses a different algorithm;
- ❖ Goal: to yield robust clustering without human interaction and/or parameter tuning.

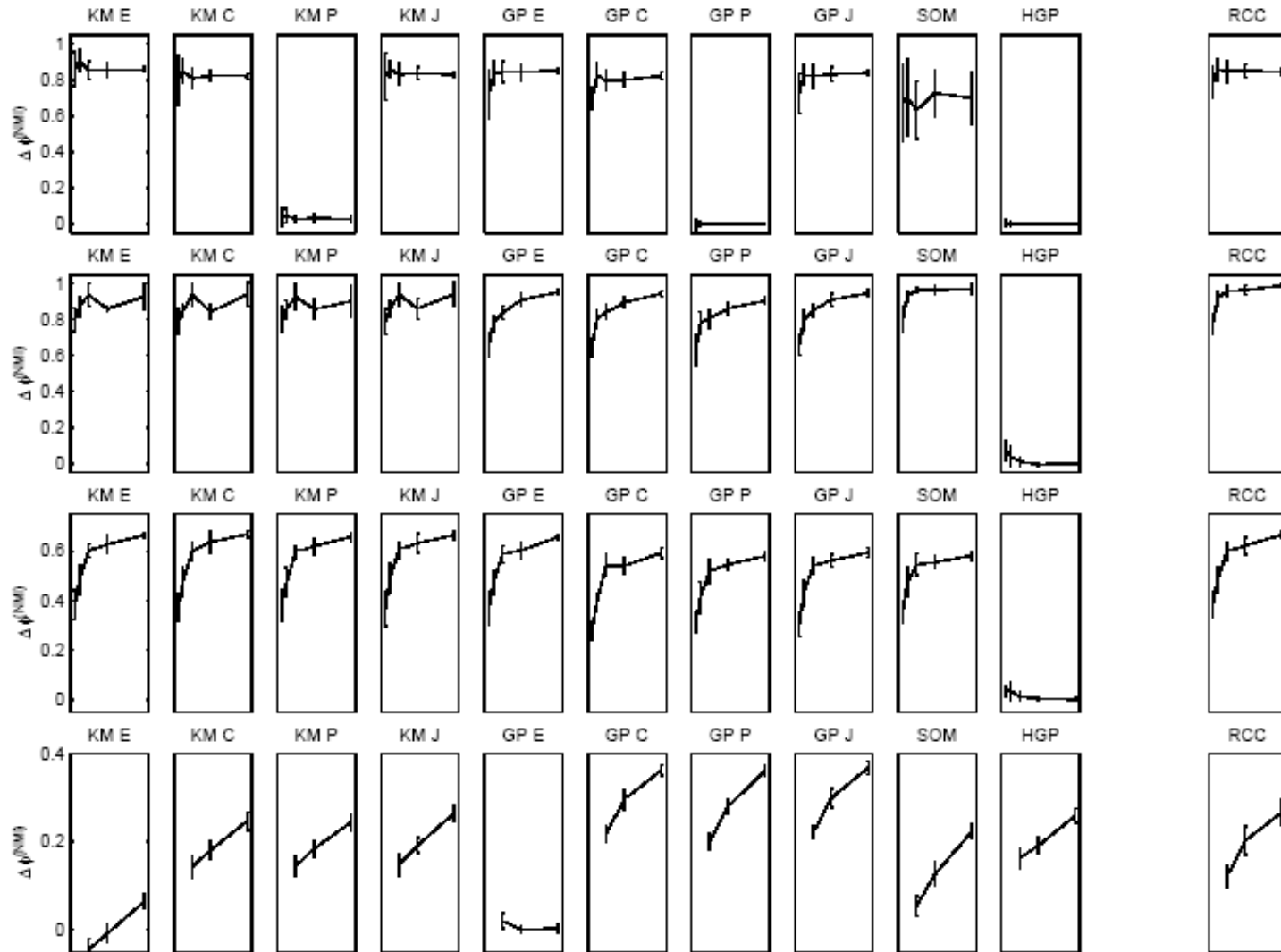
RCC: Settings

- ❖ 10 algorithms: (1) SOMs; (2) Hypergraph partitioning; k-means with distance based on (3) Euclidean; (4) cosine; (5) correlation; (6) extended Jaccard; and graph partitioning with distance based on (7) Euclidean; (8) cosine; (9) correlation; (10) extended Jaccard.
- ❖ Each clusterer uses a different algorithm;
- ❖ Goal: to yield robust clustering without human interaction and/or parameter tuning.

RCC: Evaluation

- ❖ RCC was performed 10 times each on sample sizes of 50, 100, 200, 400, 800 (200, 400, and 800 for YAHOO).
- ❖ Quality is measured in terms of difference in mutual information as compared to a random clustering algorithm.

RCC: Results



RCC: Remarks

- ❖ Cluster ensembles can be used to boost robustness in risk-intolerant settings;
- ❖ It's hard to evaluate clustering in high-dimensional settings. Category labels may not be available. One can:
 - ❖ Utilize several algorithms;
 - ❖ Integrate their results using a consensus function.

Concluding Remarks

- ❖ Is consensus clustering robust with respect to different values of k that may be returned by different clusterings?
- ❖ ANMI (unsupervised consensus function) seems a good indicator of NMI with the “true labels” (may not be available!)
- ❖ Apply ANMI to allow soft clustering.

Additional Comments?