IEEE ICDM 2011
December 11–14, 2011

A Fast and Flexible Clustering Algorithm Using Binary Discretization

Mahito SUGIYAMA[†,‡], Akihiro YAMAMOTO[†]

[†]Kyoto University

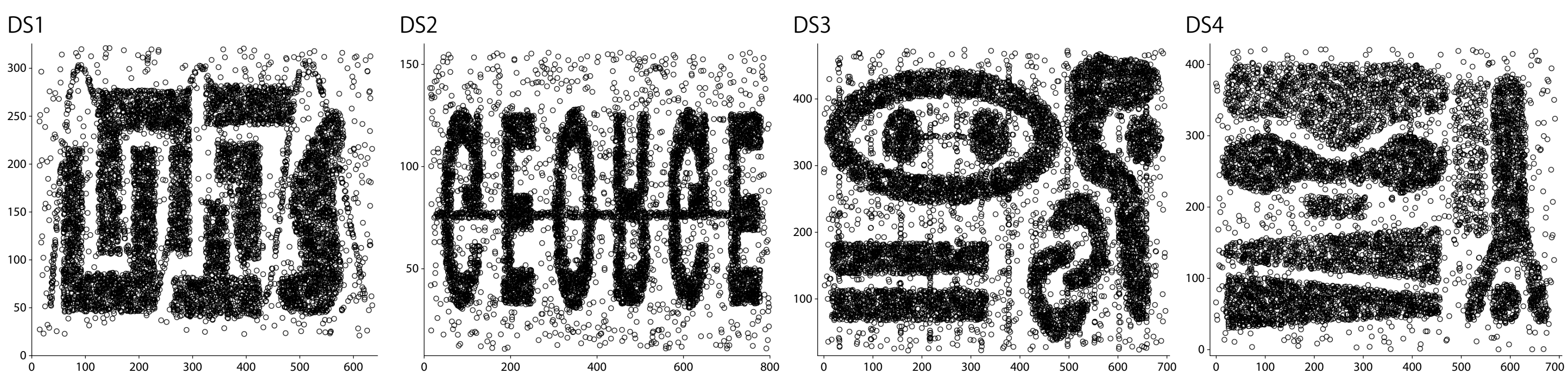
[‡]JSPS Research Fellow

Main Results

1. ***BOOL*** ***(Bianry cOding Oriented cLustering)***
    - A clustering algorithm for multivariate data to detect arbitrarily shaped clusters
    - Noise tolerant
    - Robust to changes in input parameters
2. ***BOOL is faster than K-means and the fastest among the algorithms to detect arbitrarily shaped clusters***
    - It is about two to three orders of magnitude faster than two state-of-the-art algorithms [Chaoji *et al*. SDM 2011, Chaoji *et al*. KAIS 2009] that can detect non-convex clusters of arbitrary shapes

Evaluation with Typical Benchmarks

- We used four synthetic databases (DS1 - DS4)
  - From the CLUTO website
    - `http://glaros.dtc.umn.edu/gkhome/views/cluto/`
- Typical benchmarks for (spatial) clustering
  - CHAMELEON [Karypis *et al.*, 1999], SPARCL [Chaoji *et al.*, 2009], ABACUS [Chaoji *et al.*, 2011], and so on



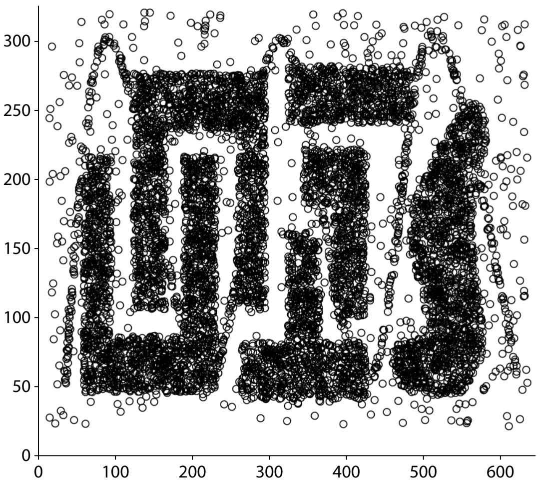
Results for Benchmarks

- Running time (in seconds)
  - In table, n denotes the number of data points
  - Results for ABACUS and SPARCL are from [Chaoji *et al.*, 2011]
    - Their source codes are not publicly available

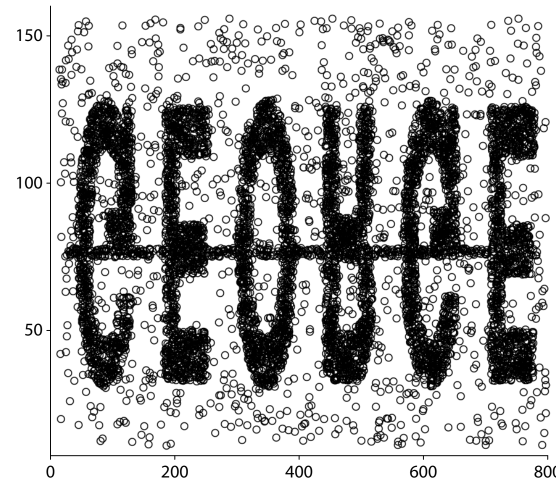
		Non-convex				Convex
	n	BOOL	DBSCAN	ABACUS	SPARCL	K-means
DS1	8000	0.004	9.959	1.7	1.8	0.008
DS2	8000	0.004	10.041	1.3	1.5	0.008
DS3	10000	0.010	15.832	1.9	2.5	0.036
DS4	8000	0.005	9.947	1.7	1.8	0.018

Results (Original Data)

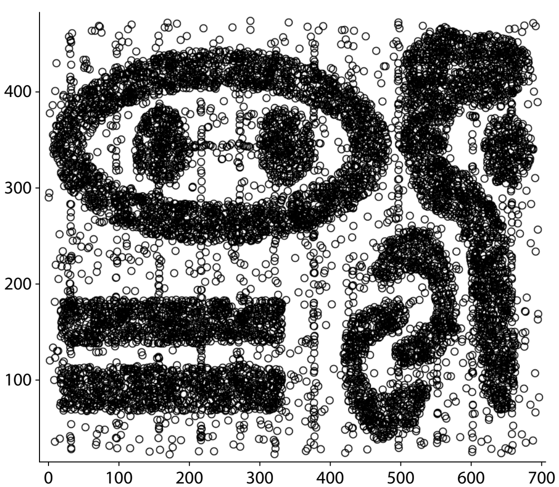
DS1



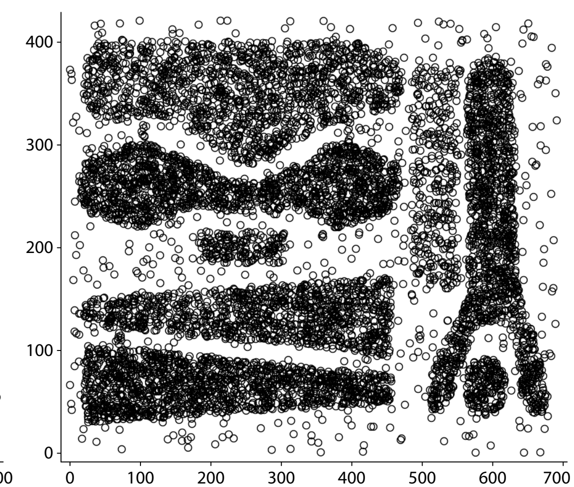
DS2



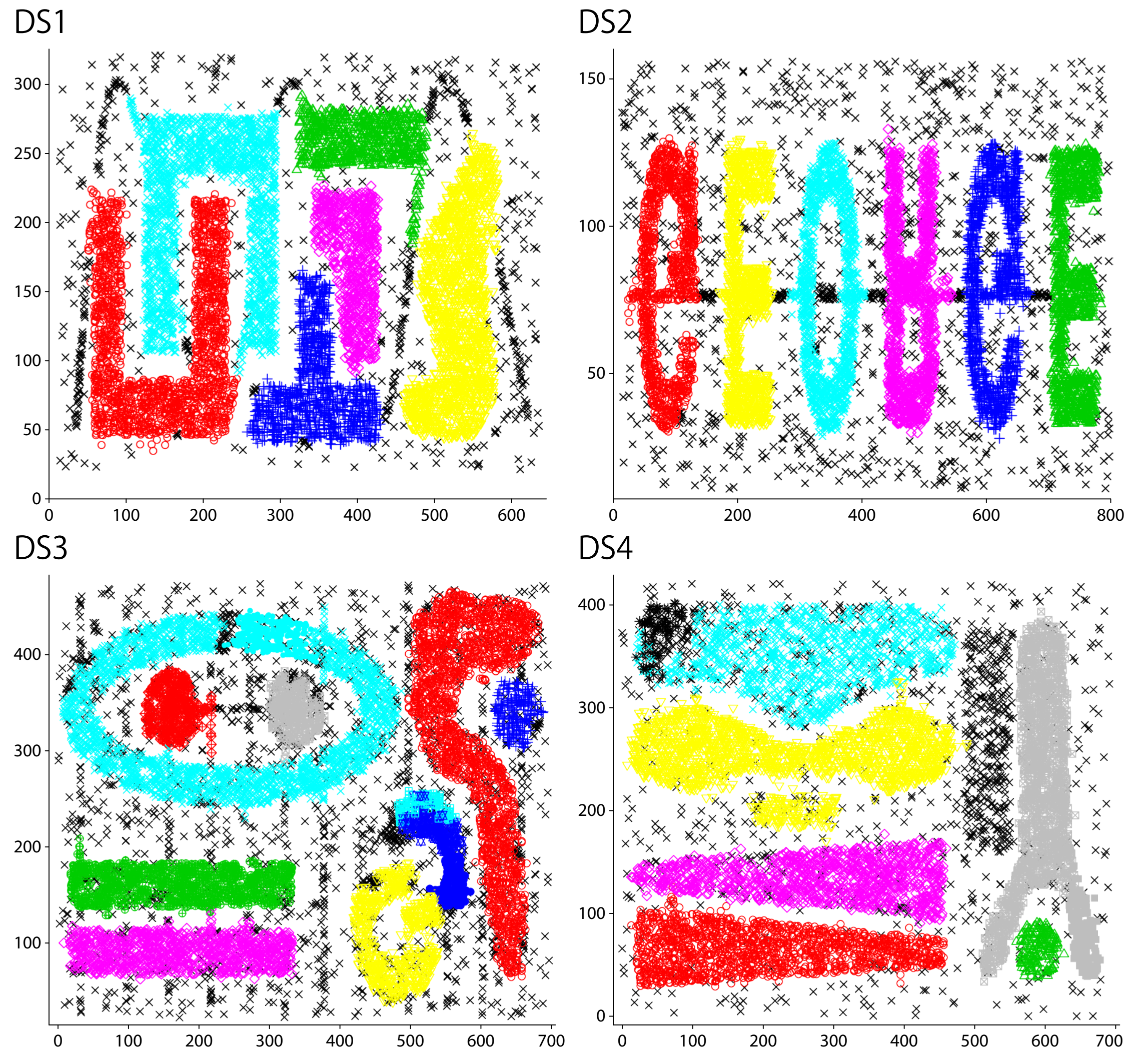
DS3



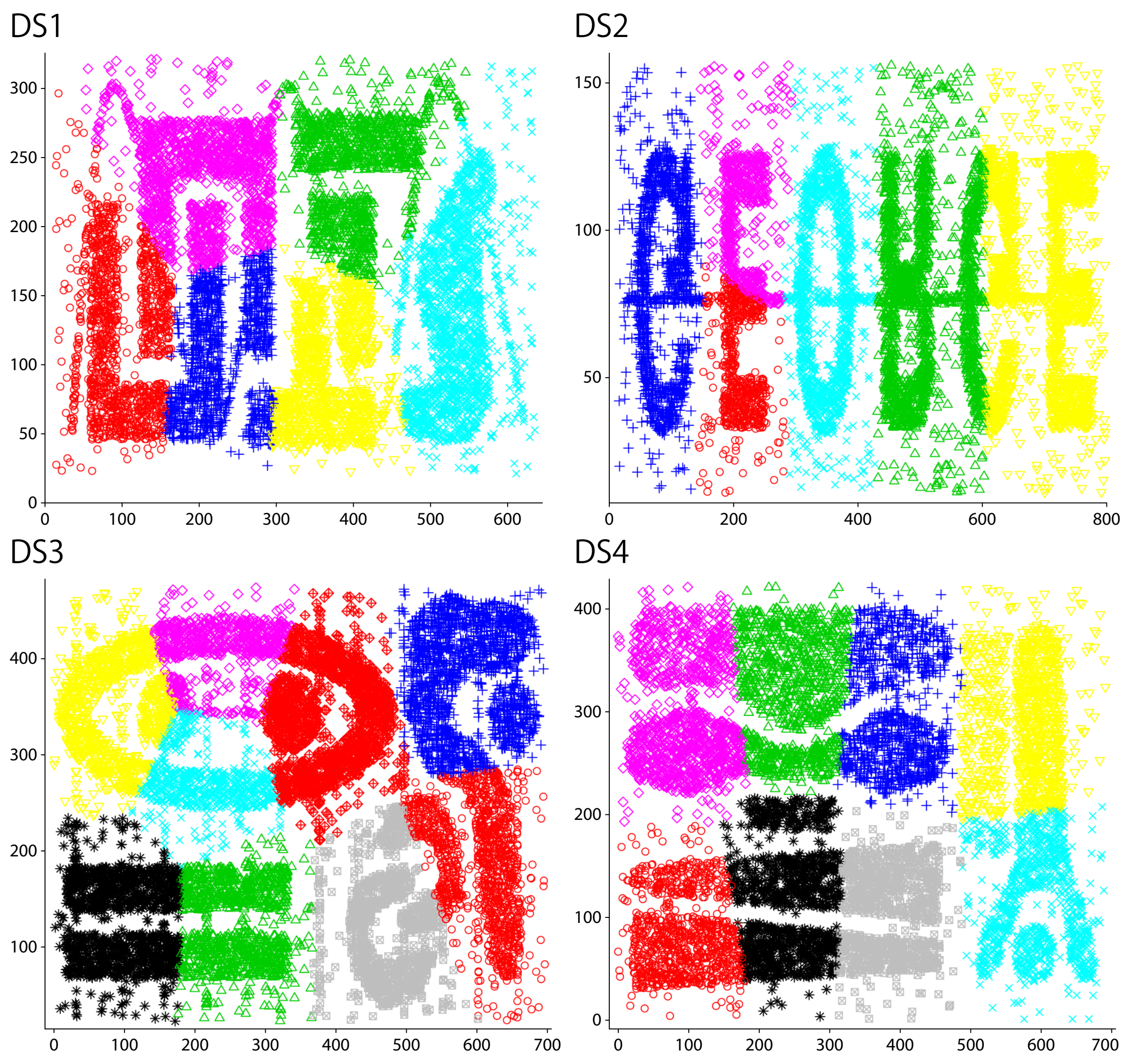
DS4



Results (BOOL)



Results (*K*-means)



Evaluation with Natural Images

- We used four natural images
  - From the Berkeley segmentation database and benchmark
    - `http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/`
- 481×321 in size, 154,401 pixels in total
  - The RGB (red-green-blue) values for each pixel were obtained by pre-precessing, same as [Chaoji *et al.*, 2011]

Horse

Mushroom

Pyramid

Road

Results for Natural Images

- Running time (in seconds)
  - In table, n denotes the number of data points
  - Results for ABACUS and SPARCL are from [Chaoji *et al.*, 2011]

		Non-convex			Convex
	n	BOOL	ABACUS	SPARCL	*K*means
Horse	154401	0.253	31.2	41.8	0.674
Mushroom	154401	0.761	29.3	–	1.449
Pyramid	154401	0.187	11.3	–	0.254
Road	154401	0.180	14.9	–	0.209

Results (Original Data)

Horse

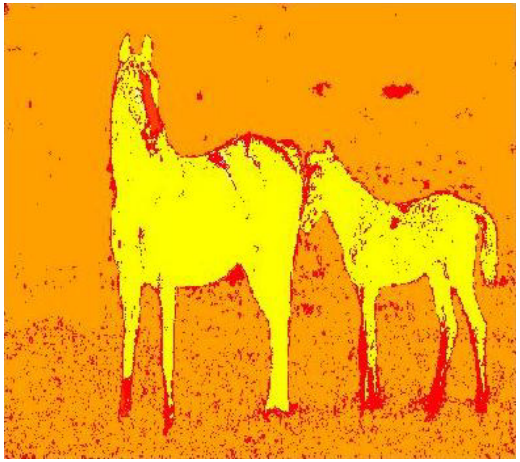
Mushroom

Pyramid

Road

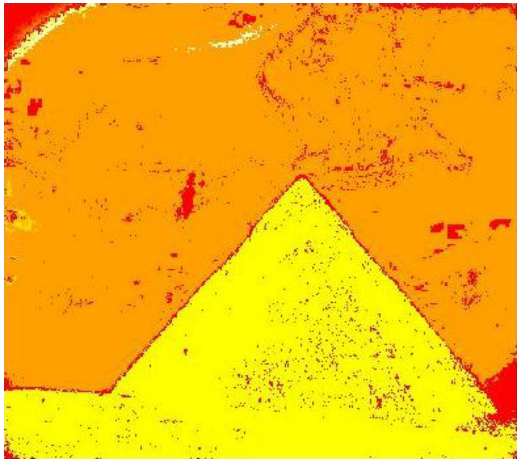
Results (BOOL)

Horse

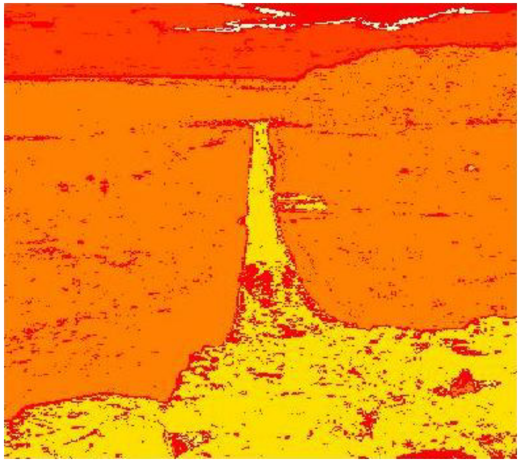


Mushroom

Pyramid

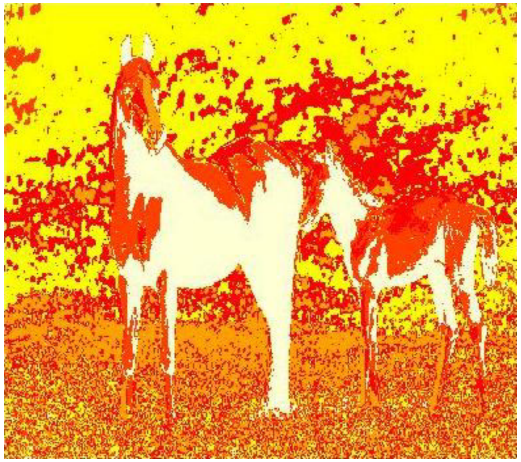


Road

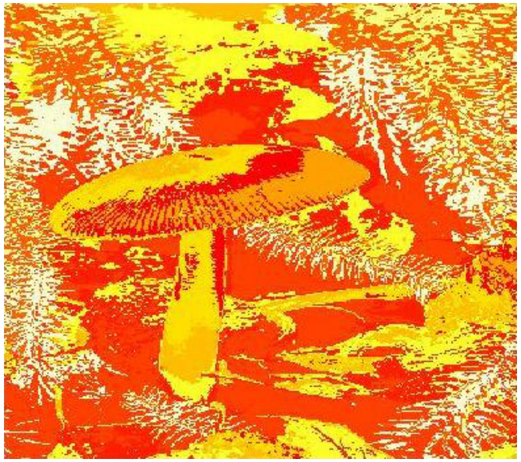


Results (*K*-means)

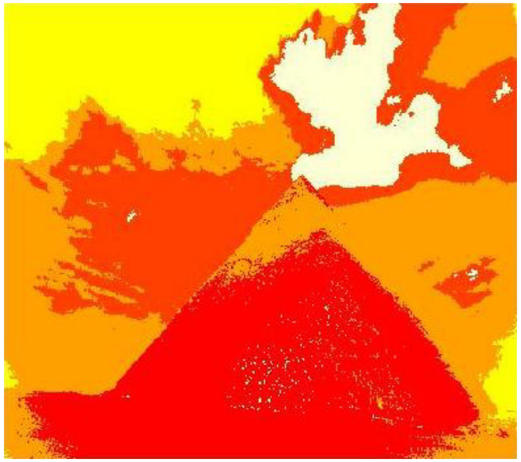
Horse



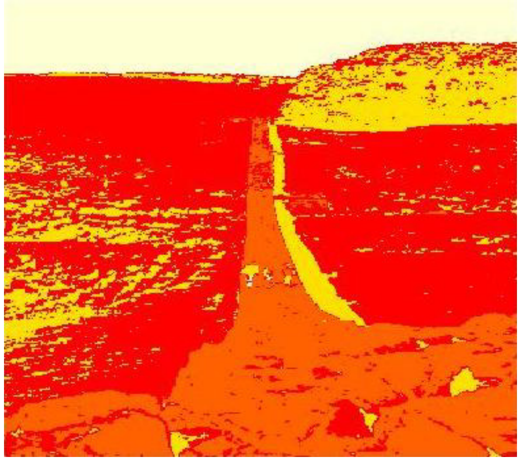
Mushroom



Pyramid



Road



Results for UCI Data

- Running time (in seconds)

	n	d	K	BOOL	K-means	DBSCAN
ecoli	336	7	8	0.001	0.002	0.111
sonar	208	60	2	0.004	0.005	0.149
shuttle	14500	9	7	0.025	0.065	–
wdbc	569	30	2	0.004	0.004	0.222
wine	178	13	3	0.002	0.002	0.047
wine quality	4898	11	7	0.019	0.026	7.601

Results for UCI Data

- Adjusted Rand index

	n	d	K	BOOL	K-means	DBSCAN
ecoli	336	7	8	0.5745	0.4399	0.1223
sonar	208	60	2	0.0133	0.0064	0.0006
shuttle	14500	9	7	0.7651	0.1432	–
wdbc	569	30	2	0.6806	0.4914	0.5530
wine	178	13	3	0.4638	0.3347	0.2971
wine quality	4898	11	7	0.0151	0.0099	0.0134

Motivation

- The *K*-means algorithm is widely used
  - Simple and efficient
  - The main drawback is its limited clustering ability: Non-spherical clusters cannot be found
- Many shape-based algorithms have been proposed
  - Can find clusters of arbitrary shape
  - Drawbacks:
    1. Not scalable (time complexity is quadratic or cubic)
    2. Not robust to input parameters

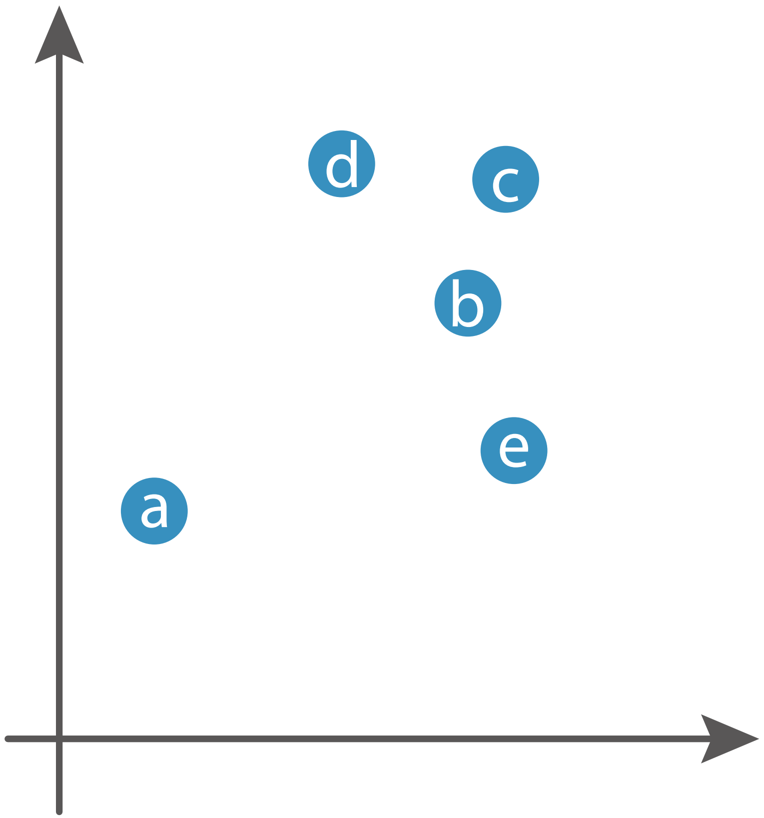
Key Strategy

- Requirements:
  1. Fast (linear in data size)
  2. Robust
  3. Find arbitrary shaped clusters

- **Key idea**: Translate input data onto binary words using binary discretization
  - A hierarchy of clusters is naturally produced by increasing the accuracy of the discretization
- Each cluster is constructed by agglomerating adjacent smaller clusters
  - performed linearly by sorting binary representations

Clustering Process

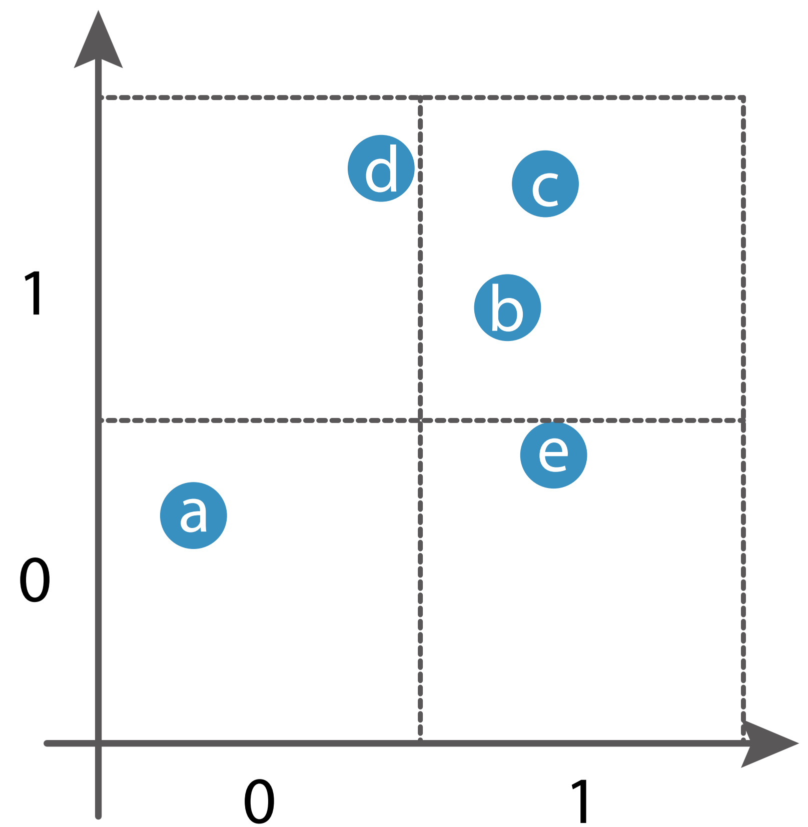
id	x	y
a	0.14	0.31
b	0.66	0.71
c	0.72	0.86
d	0.48	0.89
e	0.74	0.48



Discretize Database at Level 1

Discretization level: 1

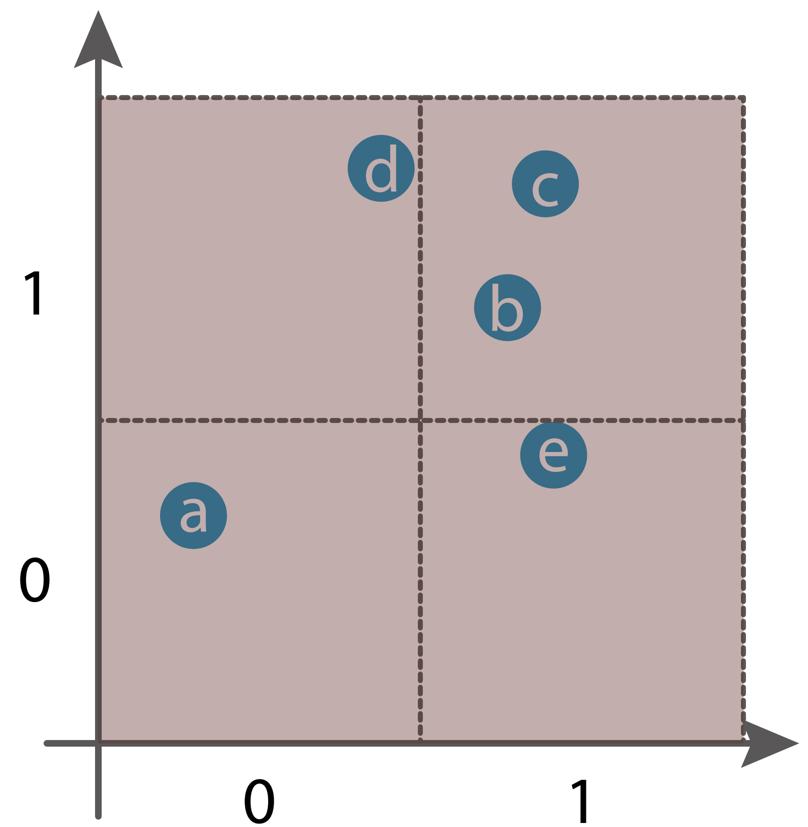
id	x	y
a	0.14	0.31
b	0.66	0.71
c	0.72	0.86
d	0.48	0.89
e	0.74	0.48



Discretize Database at Level 1

Discretization level: 1

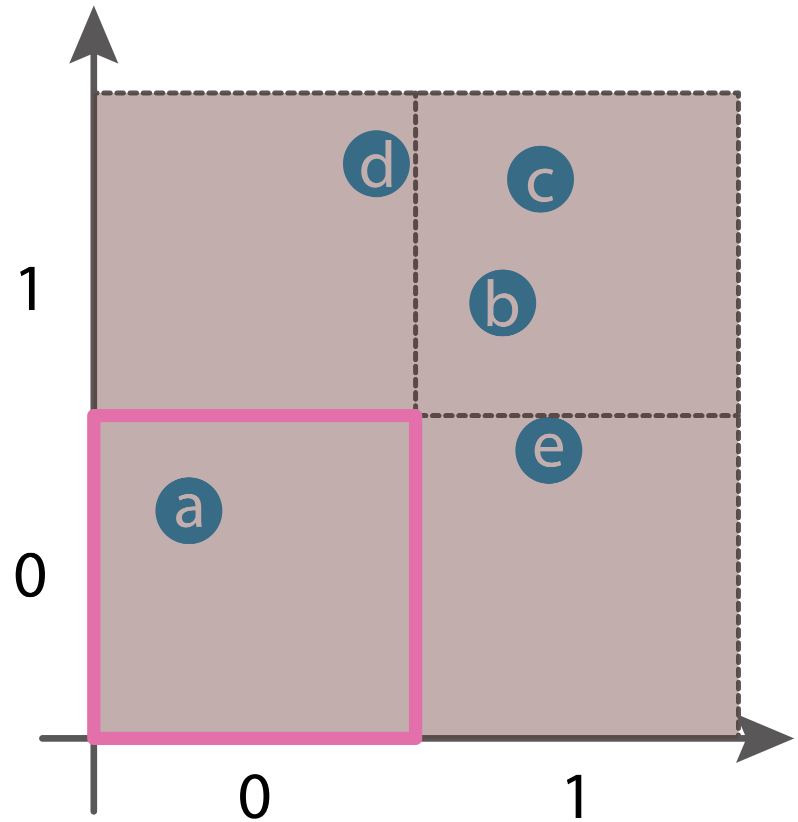
id	x	y
a	0	0
b	1	1
c	1	1
d	0	1
e	1	0



Discretize Database at Level 1

Discretization level: 1

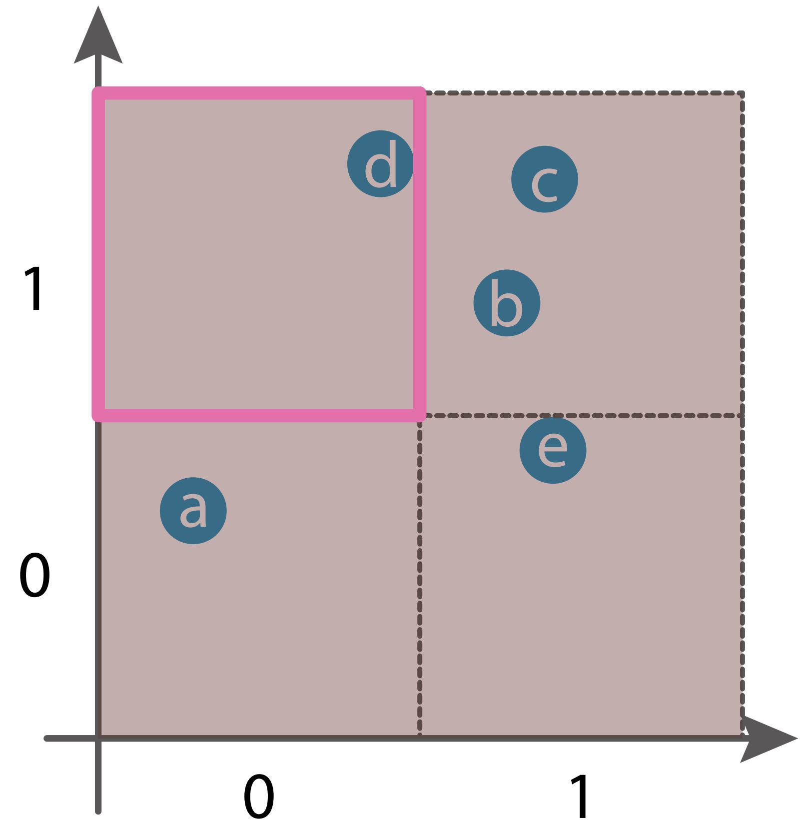
id	x	y
a	0	0
b	1	1
c	1	1
d	0	1
e	1	0



Discretize Database at Level 1

Discretization level: 1

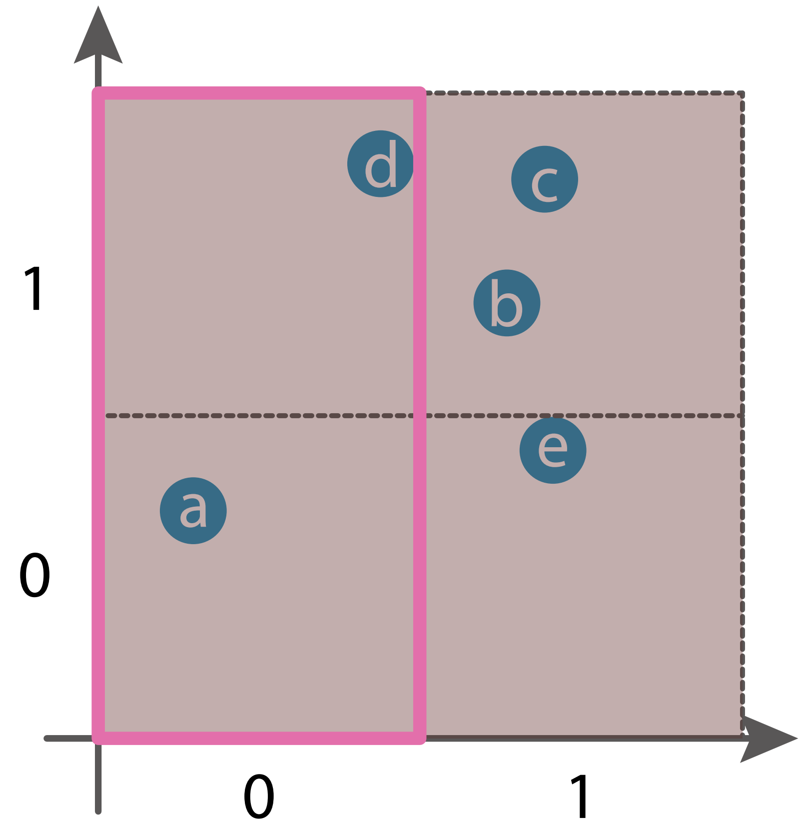
id	x	y
a	0	0
b	1	1
c	1	1
d	0	1
e	1	0



Discretize Database at Level 1

Discretization level: 1

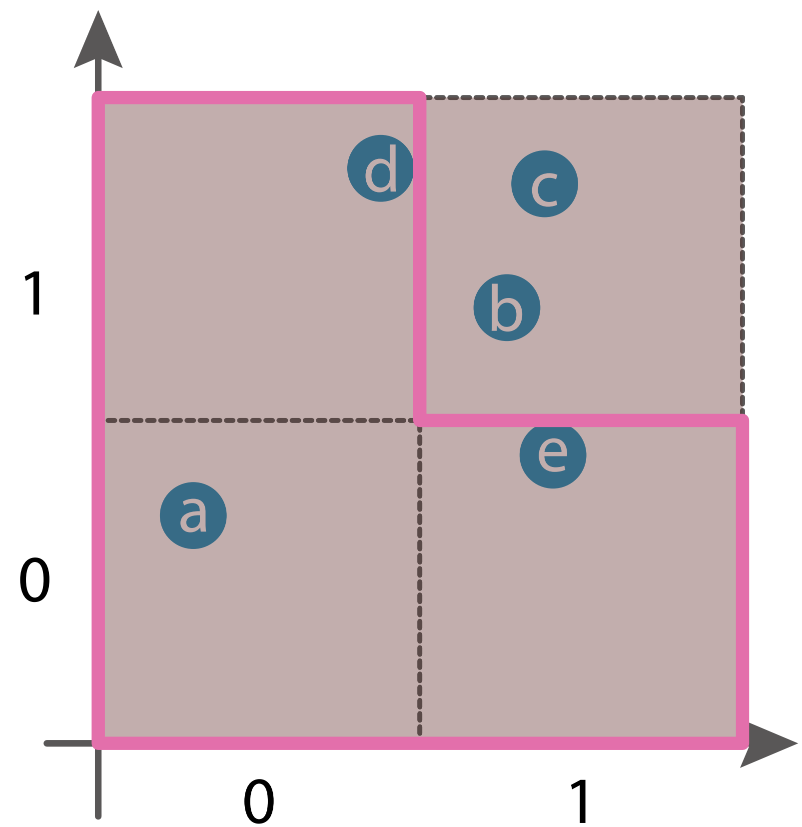
id	x	y
a	0	0
b	1	1
c	1	1
d	0	1
e	1	0



Discretize Database at Level 1

Discretization level: 1

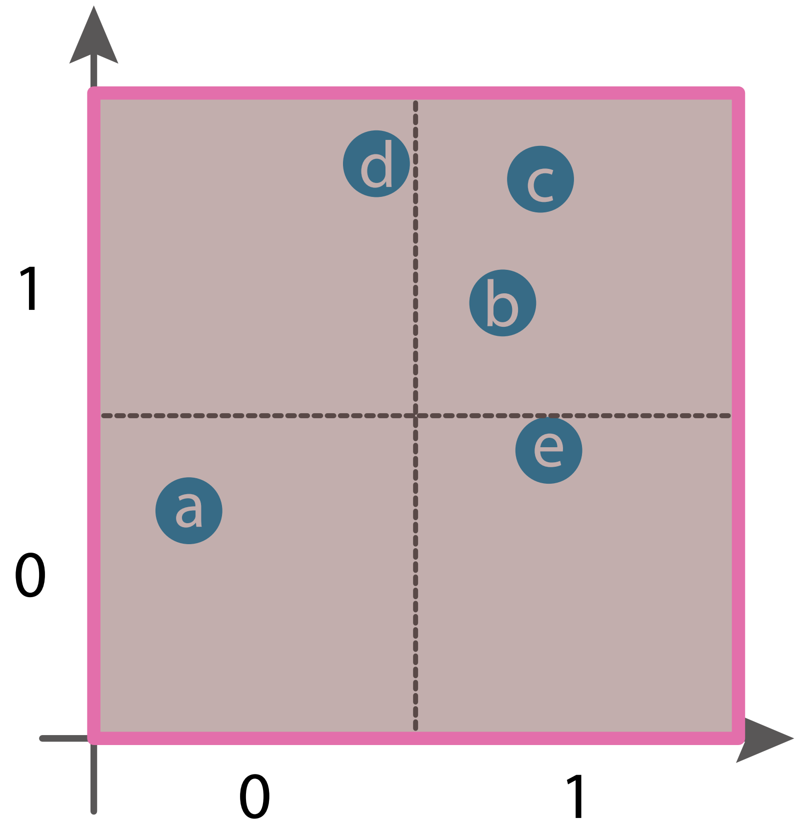
id	x	y
a	0	0
b	1	1
c	1	1
d	0	1
e	1	0



Agglomerate Adjacent Clusters

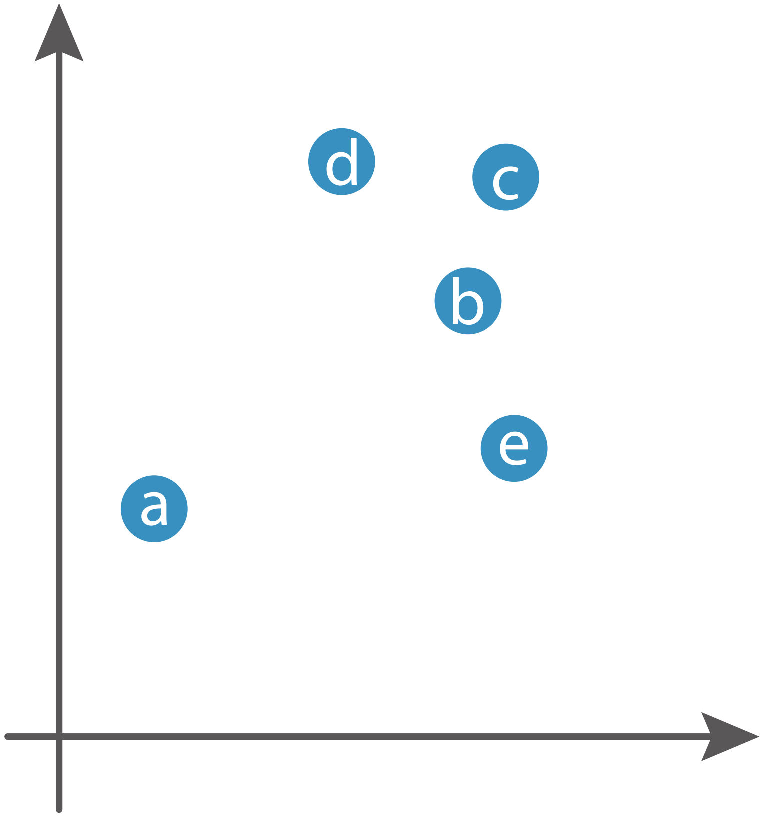
Discretization level: 1

id	x	y
a	0	0
b	1	1
c	1	1
d	0	1
e	1	0



Go to Next Level

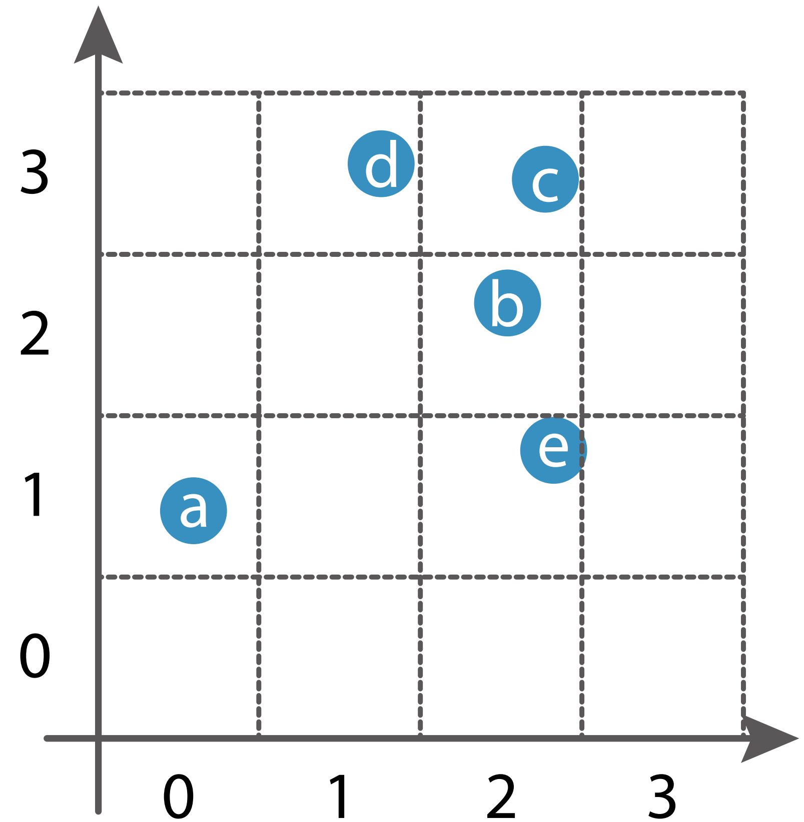
id	x	y
a	0.14	0.31
b	0.66	0.71
c	0.72	0.86
d	0.48	0.89
e	0.74	0.48



Discretize Database at Level 2

Discretization level: 2

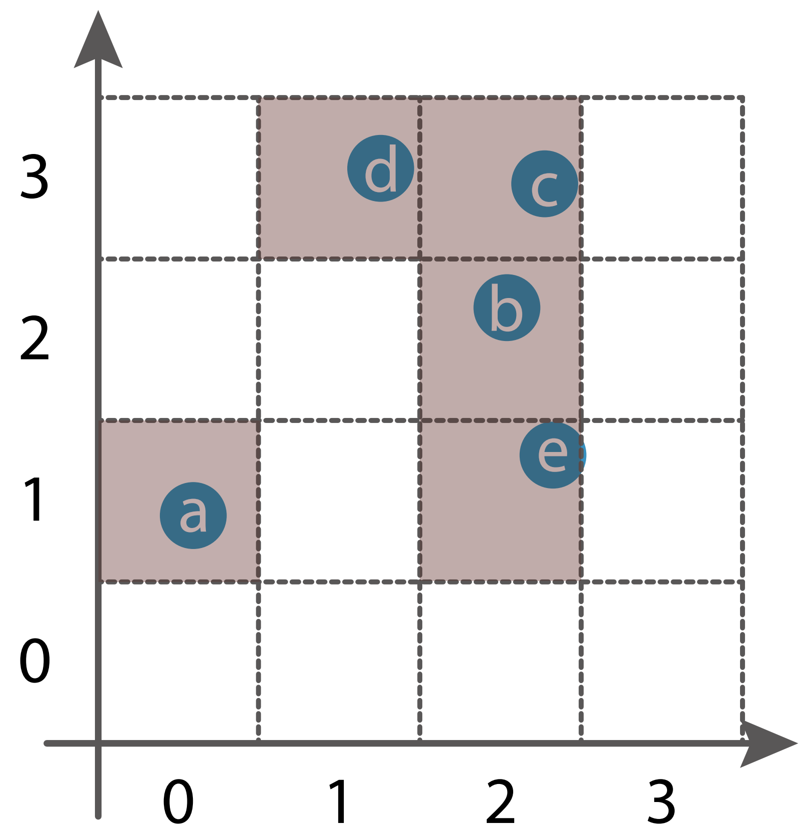
id	x	y
a	0	1
b	2	2
c	2	3
d	1	3
e	2	1



Discretize Database at Level 2

Discretization level: 2

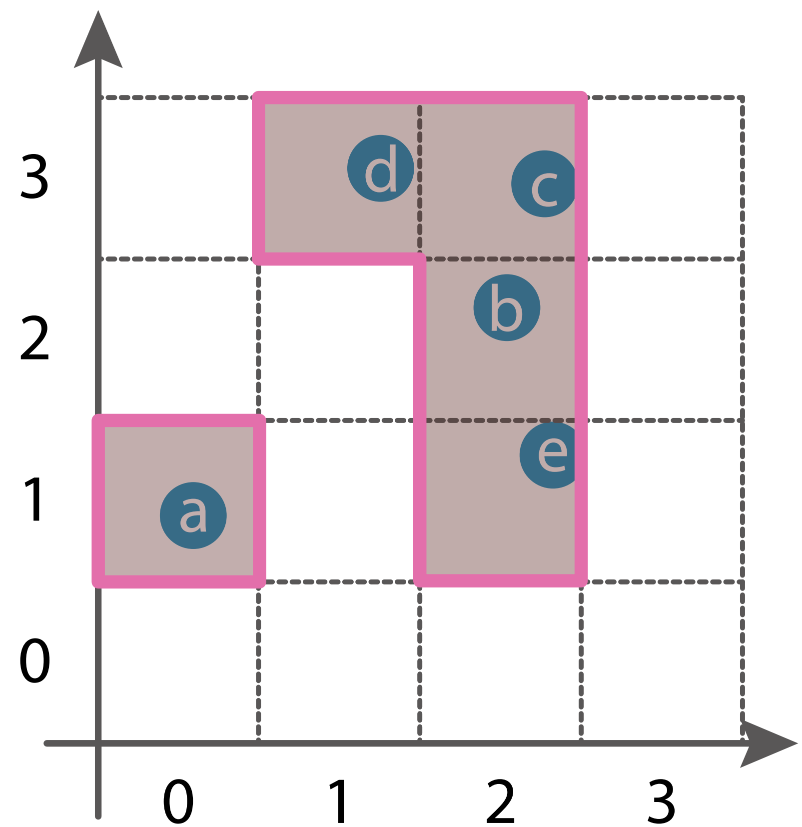
id	x	y
a	0	1
b	2	2
c	2	3
d	1	3
e	2	1



Agglomerate Adjacent Clusters

Discretization level: 2

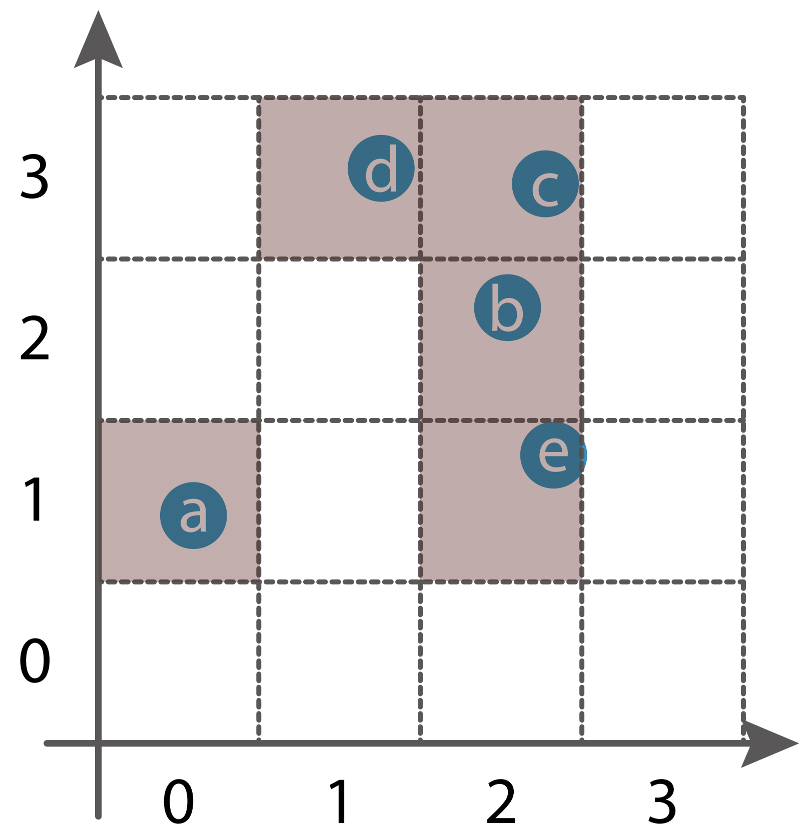
id	x	y
a	0	1
b	2	2
c	2	3
d	1	3
e	2	1



Construct Clusters with Sorting

Discretization level: 2

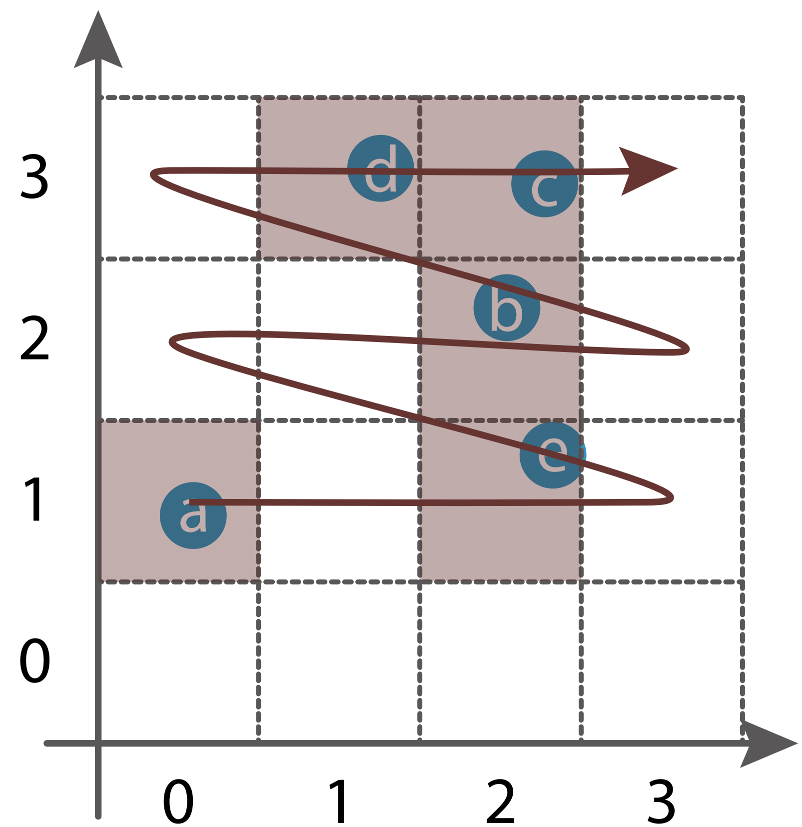
id	x	y
a	0	1
b	2	2
c	2	3
d	1	3
e	2	1



Sort by *x*-axis → Sort by *y*-axis

Discretization level: 2

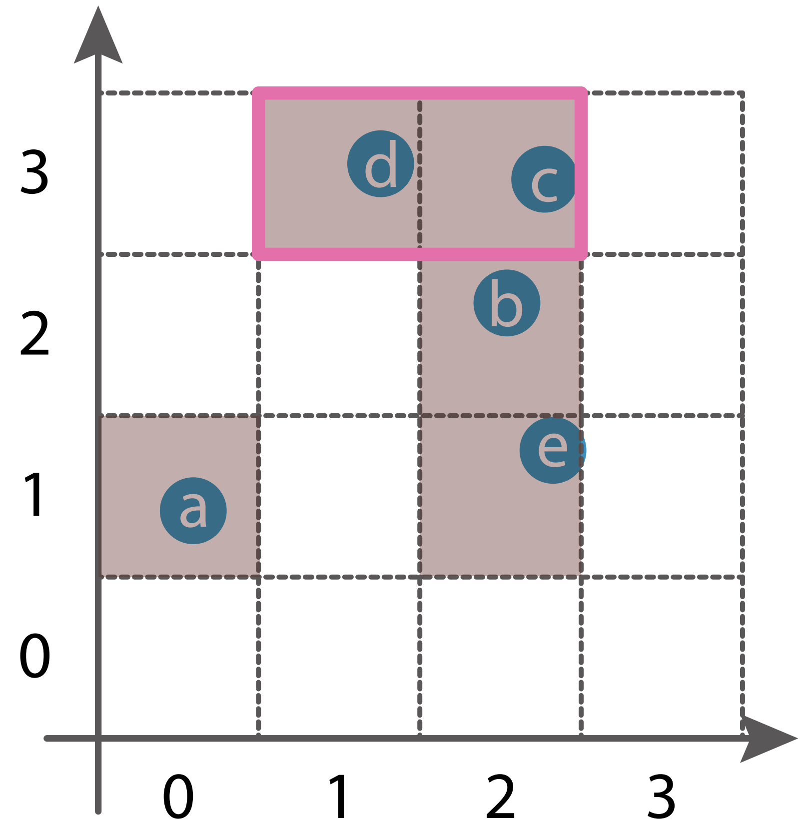
id	*x*	*y*
a	0	1
e	2	1
b	2	2
d	1	3
c	2	3



Compare Each Point to the Next Point

Discretization level: 2

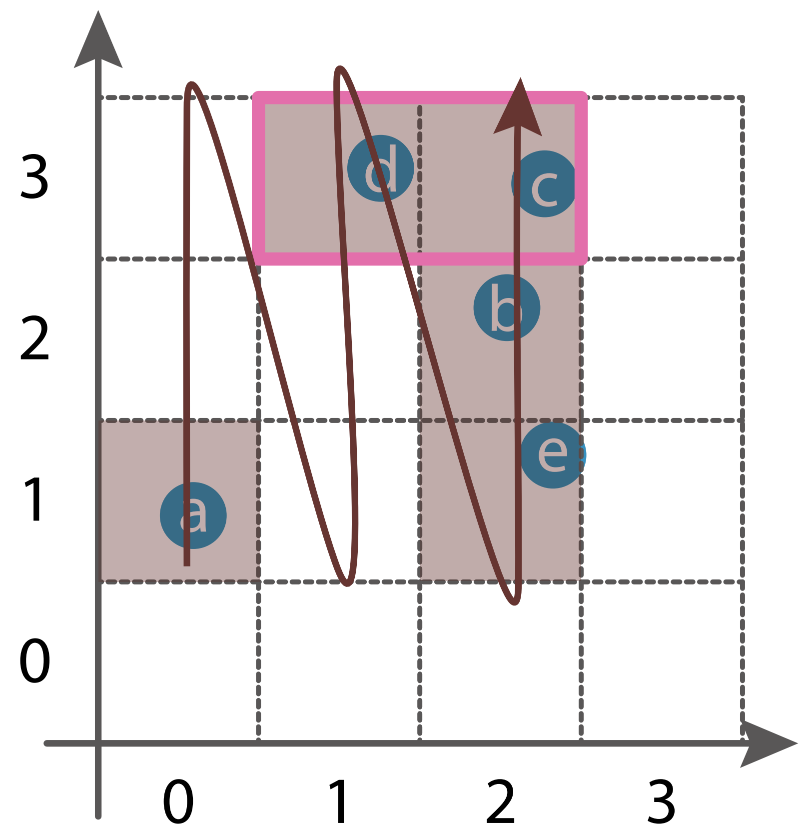
id	x	y
a	0	1
e	2	1
b	2	2
d	1	3
c	2	3



Sort by *x*-axis

Discretization level: 2

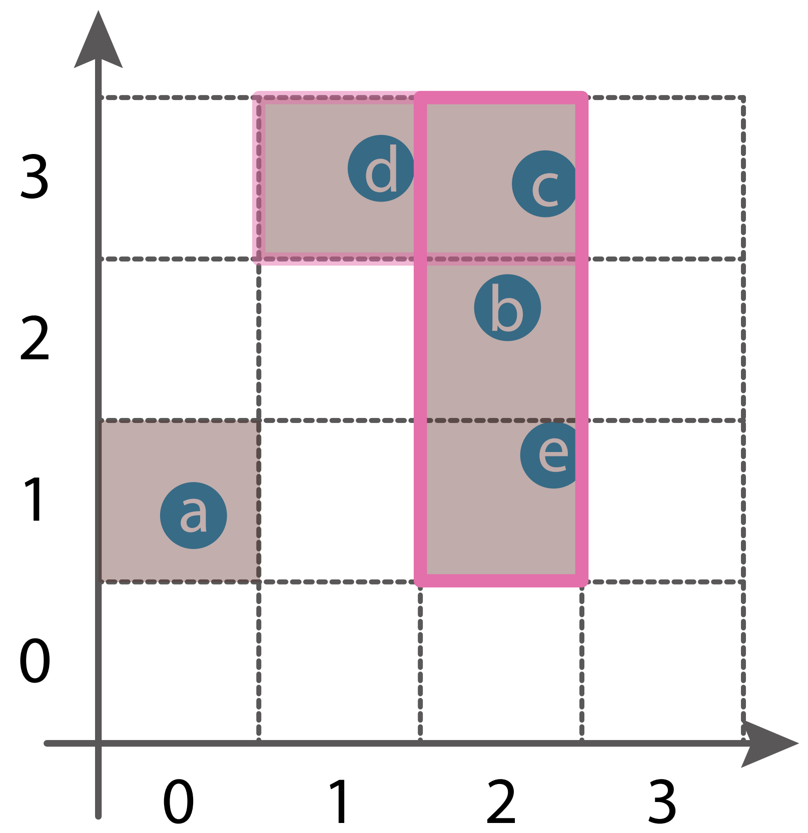
id	x	y
a	0	1
d	1	3
e	2	1
b	2	2
c	2	3



Compare Each Point to the Next Point

Discretization level: 2

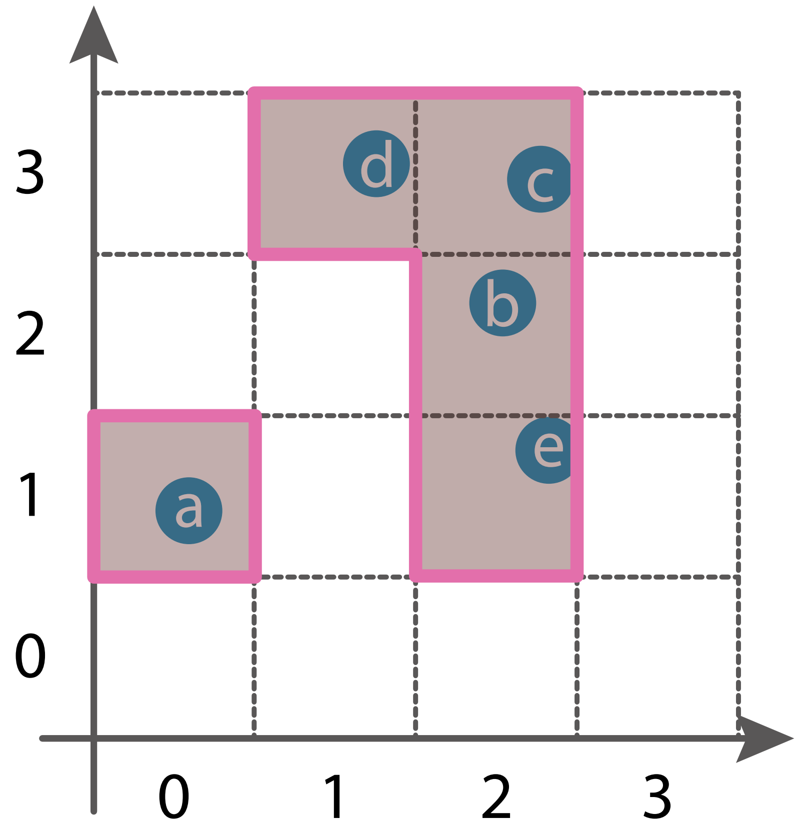
id	x	y
a	0	1
d	1	3
e	2	1
b	2	2
c	2	3



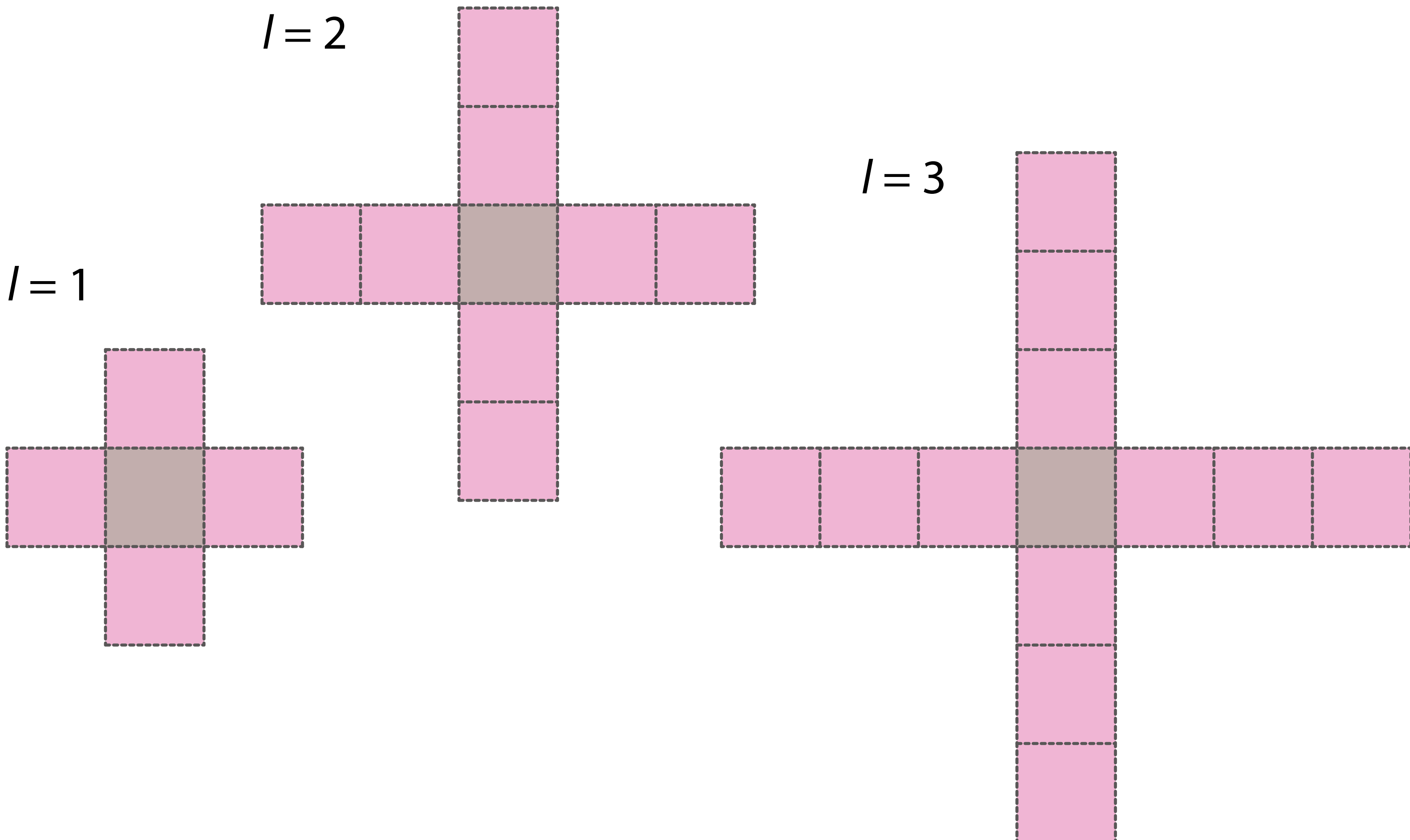
Finish (... or go to the next level)

Discretization level: 2

id	x	y
a	0	1
d	1	3
e	2	1
b	2	2
c	2	3



Distance Parameter l



Noise Filtering by BOOL

- Noise filtering is important in clustering
- Define $\mathscr{C}_{\geqslant N} := \{C \in \mathscr{C} \mid \#C \geqslant N\}$ for a partition $\mathscr{C}$
  - See a cluster C as noises if $\#C < N$
- Example: Given $\mathscr{C} = \{\{0.1\}, \{0.4, 0.5, 0.6\}, \{0.9\}\}$
  - $\mathscr{C}_{\geqslant 2} = \{\{0.4, 0.5, 0.6\}\}$, and 0.1 and 0.9 are noises
- We input the number (noise parameter) N as the lower bound of the size of cluster in BOOL

Conclusion

- We have developed a clustering algorithm BOOL
  - Fastest w.r.t finding arbitrarily shaped clusters
  - Robust, highly scalable, and noise tolerant
- The key to performance is discretization based on the binary encoding and sorting of data points
- Future work: BOOL is simple and can easily be extended to other machine learning tasks
  - *e.g.*, anomaly detection, semi-supervised learning

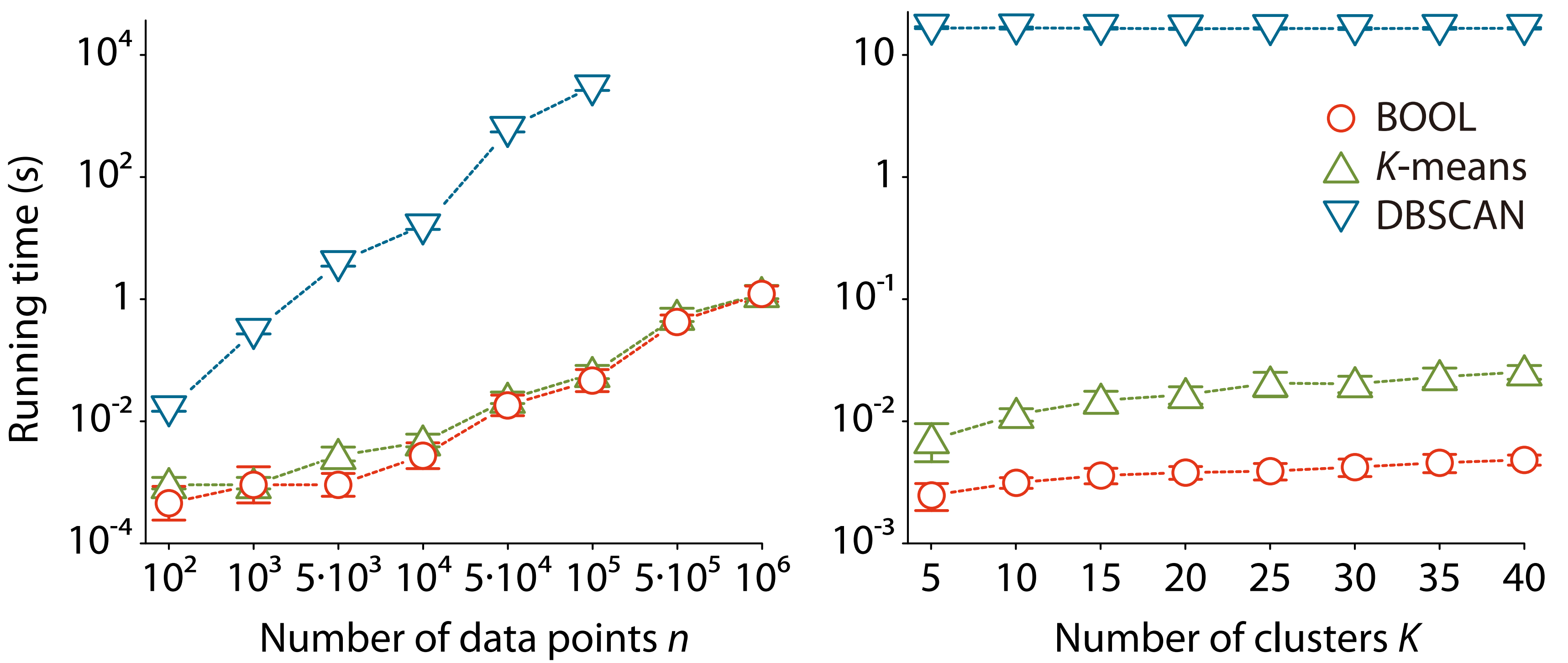
Appendix

Other Experiments

- We evaluate BOOL experimentally to determine its scalability and effectiveness for various types of databases
  - Randomly generated synthetic databases ($d = 2$)
  - Famous synthetic databases for spatial clustering ($d = 2$)
  - Real databases generated from natural images ($d = 3$)
  - Real databases from the UCI repository ($d = 7, 9, 30, 60$)
- We used Mac OS X version 10.6.5 (2×2.26-GHz Quad-Core Intel Xeon CPUs, 12 GB of memory)
  - BOOL was implemented in C, compiled with gcc 4.2.1
  - All experiments were performed in R version 2.12.2
- All databases are pre-processed by min-max normalization in BOOL

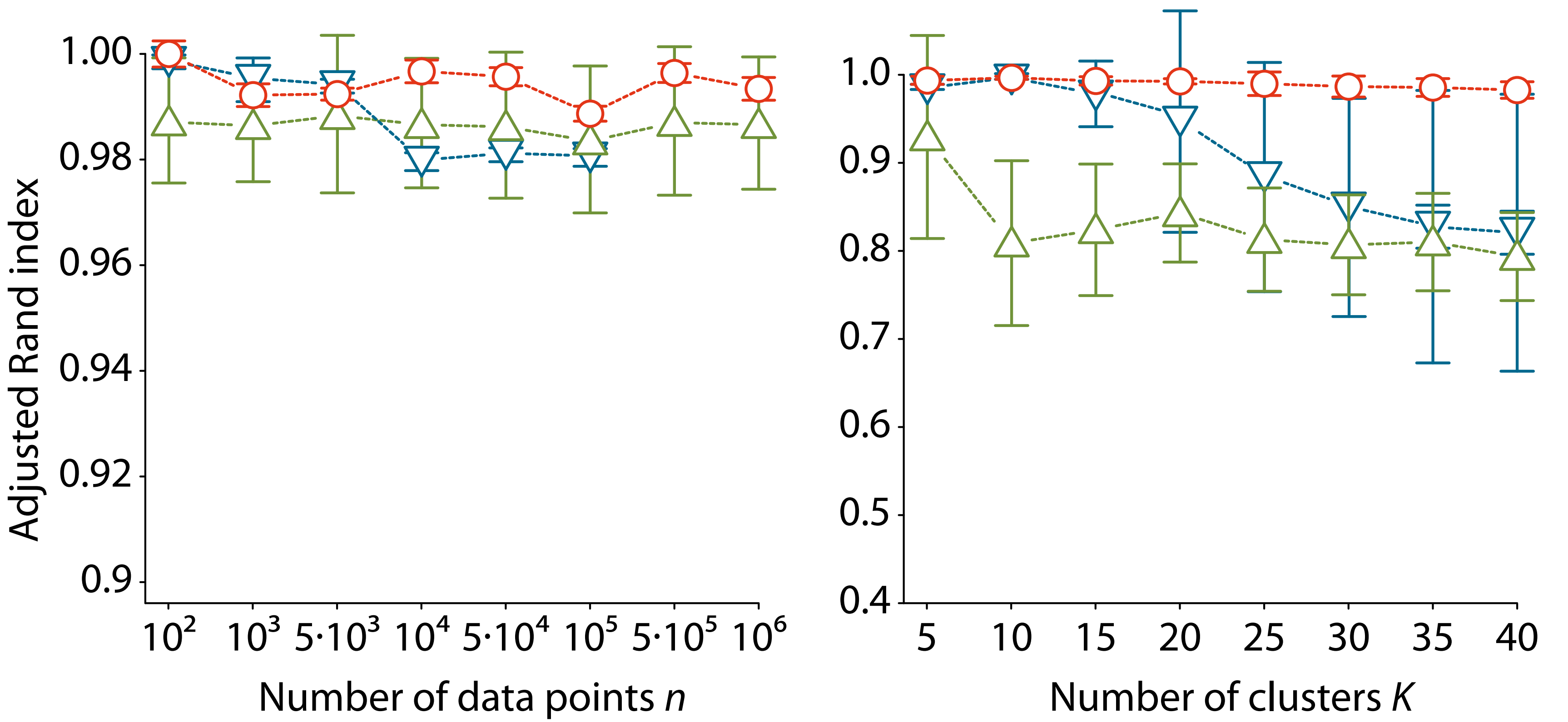
Speed w.r.t. # Data and # Clusters

- Randomly generated synthetic databases
  - $K = 5$ (left), $n = 10000$ (right)



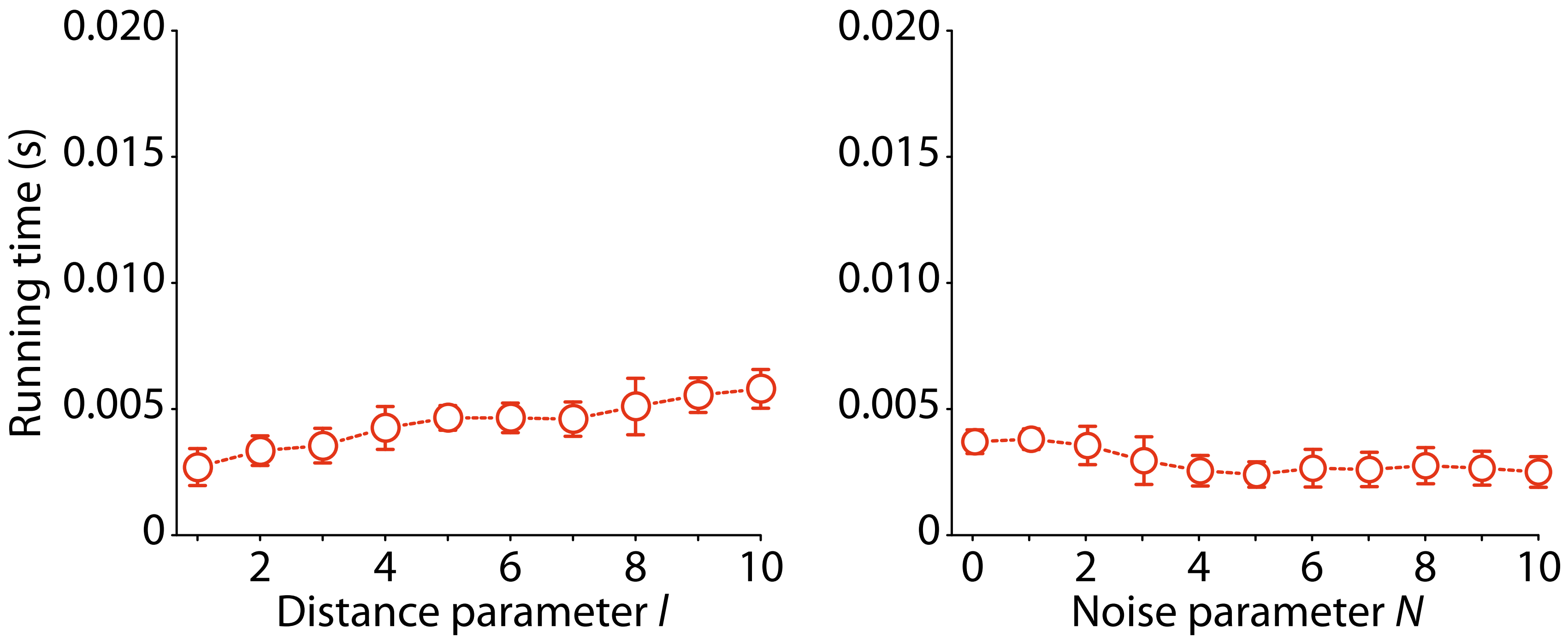
Quality w.r.t. # Data and # Clusters

- Randomly generated synthetic databases
  - K = 5 (left), n = 10000 (right)



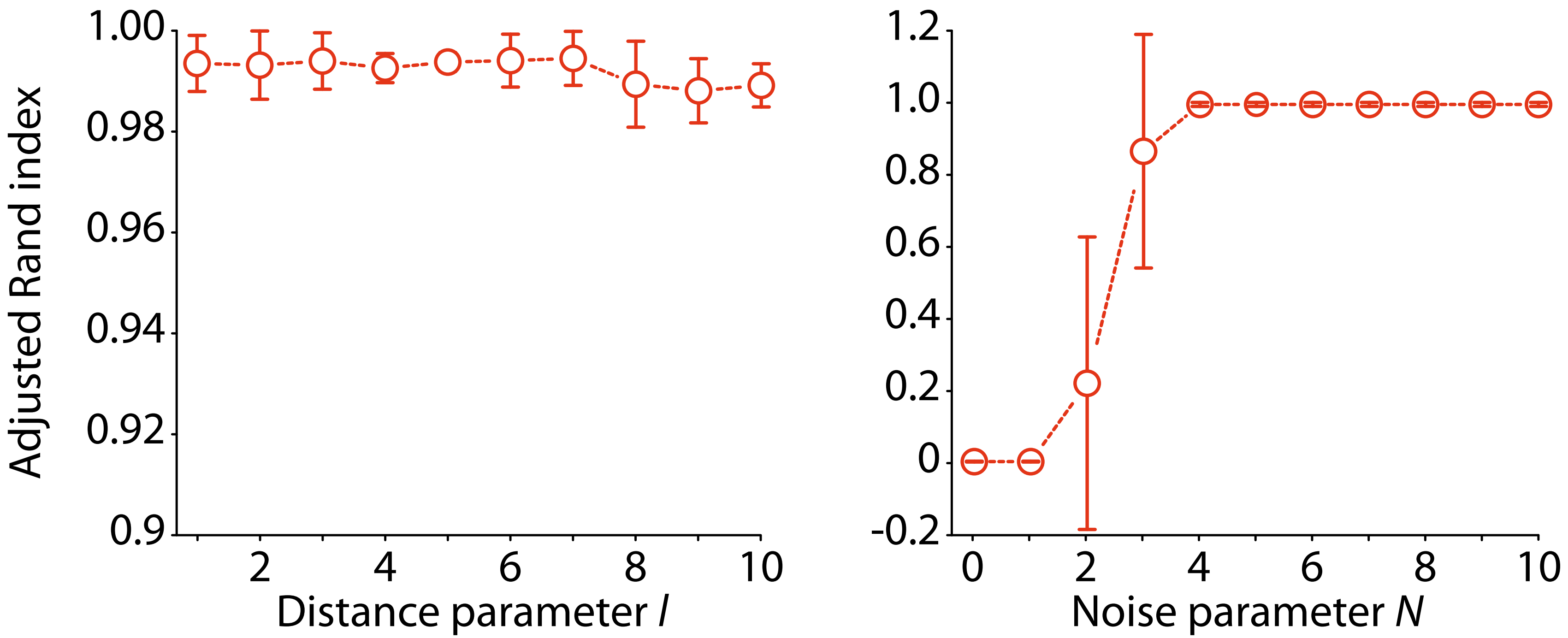
Speed w.r.t. Input Parameters

- Randomly generated synthetic databases
  - $K = 5$ and $n = 10000$



Quality w.r.t. Input Parameters

- Randomly generated synthetic databases
  - $K = 5$ and $n = 10000$



Notation

- We treat a target database as a relation [Garcia-Molina *et al.*, 2008]
  - The columns are called *attributes*
  - The domain of each attribute is $[0, 1]$
  - Attributes are composed of natural numbers $\{1, 2, \dots, d\}$
    - Each tuple corresponds to a data point in the d-dimensional Euclidean space $\mathbb{R}^d$
  - x_i is the value for attribute i of x
  - For $M \subseteq \{1, 2, \dots, d\}$, $\pi_M(X)$ denotes a database that has only the columns for attributes M of X
- Clustering is partition of X into K subsets (clusters) $C_1, \dots, C_K$
  - $C_i \neq \emptyset$ and $C_i \cap C_j = \emptyset$
  - We call $\mathscr{C} = \{C_1, \dots, C_K\}$ a partition of X
  - $\mathscr{C}(X) = \{\mathscr{C} \mid \mathscr{C}$ is a partition of $X\}$

Discretization

- For a data point x in a database X, discretization at level k is an operator Δ^k for x, where each value x_i is mapped to a natural number m such that $x_i = 0$ implies $m = 0$, and $x_i \neq 0$ implies

$$m = \begin{cases} 0 & \text{if } 0 < x_i \leqslant 2^{-k}, \\ 1 & \text{if } 2^{-k+1} < x_i \leqslant 2^{-k+2}, \\ \ldots & \\ 2^k - 1 & \text{if } 2^{-k+(k-1)} < x_i \leqslant 1. \end{cases}$$

  - We use the same operator Δ^k for discretization of a database X; *i.e.*, each data point x in X is discretized to $\Delta^k(x)$ in the database $\Delta^k(X)$.

Reachability

- Given a distance parameter $l \in \mathbb{N}$, a data point x in X is reachable at level k from a data point y if there exists a chain of points $z_1, z_2, \dots, z_p$ ($p \geqslant 2$) such that $z_1 = x, z_p = y$, and the distance

$$d_0(\Delta^k(z_i), \Delta^k(z_{i+1})) \leqslant 1 \text{ and } d_\infty(\Delta^k(z_i), \Delta^k(z_{i+1})) \leqslant l$$

for all $i \in \{1, 2, \dots, p-1\}$

  – The distance between x and y is defined by

$$d_0(x, y) = \sum_{i=1}^{d} \delta(x_i, y_i), \text{ where } \delta = \begin{cases} 0 & \text{if } x_i = y_i, \\ 1 & \text{if } x_i \neq y_i, \end{cases}$$

$$d_\infty(x, y) = \max_{i \in \{1,\dots,d\}} |x_i - y_i|$$

in the L_0 and L_∞ metrics, respectively

Pseudo-Code (1/3)

Input: Database X,
lower bound on number of clusters K,
noise parameter N, and
distance parameter l

Output: Partition $\mathscr{C}$

function $\textsc{Bool}(X, K, N)$

$k \leftarrow 1$ // k is level of discretization
repeat
  $\mathscr{C} \leftarrow \textsc{MakeHierarchy}(X, k, N)$
  $k \leftarrow k + 1$
until $\#\mathscr{C} \geqslant K$
output $\mathscr{C}$

Pseudo-Code (2/3)

function $\textsc{MakeHierarchy}(X, k, N)$

$\mathscr{C} \leftarrow \{\{x\} \mid x \in X\}$
$h \leftarrow d$ // d is number of attributes of X
$X_{\mathrm{D}} \leftarrow \Delta^k(X)$ // discretize X at level k
$X \leftarrow S_{\pi_1(X_{\mathrm{D}})} \circ S_{\pi_2(X_{\mathrm{D}})} \circ \ldots \circ S_{\pi_d(X_{\mathrm{D}})}(X)$
repeat
  $X \leftarrow S_{\pi_h(X_{\mathrm{D}})}(X)$
  $\mathscr{C} \leftarrow \textsc{Aggl}(X, \mathscr{C}, h)$
  $h \leftarrow h - 1$
until $h = 0$
$\mathscr{C} \leftarrow \{C \in \mathscr{C} \mid \#C \geqslant N\}$
output $\mathscr{C}$

Pseudo-Code (3/3)

function $\textsc{Aggl}(X, \mathscr{C}, h)$

  for each object x of X

    $y \leftarrow$ successive object of x

    if $d_0(\Delta^k(x), \Delta^k(y)) \leqslant 1$ and $d_\infty(\Delta^k(x), \Delta^k(y)) \leqslant l$ **then**

      delete $C \ni x$ and $D \ni y$ from $\mathscr{C}$, and add $C \cup D$

    end if

  end for

  output $\mathscr{C}$

Level-k partition and Sorting

- BOOL construct clusters through level-k partitions ($k = 1, 2, 3, \dots$)
  - A partition of a database X is a level-k partition, denoted by $\mathscr{C}^k$, if it satisfies the following condition:
    For all pairs x and y, the pair are in the same cluster iff y is reachable at level k from x
- Sorting of a database X is defined as follows:
  - Let Y be a key database s.t. $\#X = \#Y$ and Y has only one of X's attributes
  - The expression $S_Y(X)$ is the database X for which data points are sorted in the order indicated by Y
    - Ties keep the original order of X

Properties of Reachability

- If the distance parameter $l = 1$, the condition is exactly the same as

$$d_1(\Delta^k(z_i), \Delta^k(z_{i+1})) \leqslant 1$$

  - d_1 is the Manhattan distance (L_1 metric)
- The notion of reachability is symmetric
  - If a data point x is reachable at level k from y, then y is reachable from x

Hierarchy of Clusters

- Level-k partitions have a hierarchical structure
- For the level-k and $k+1$ partitions $\mathscr{C}^k$ and $\mathscr{C}^{k+1}$, the following condition holds:
  - For every cluster $C \in \mathscr{C}^k$, there exists a set of clusters $\mathscr{D} \subseteq \mathscr{C}^{k+1}$ such that $\bigcup \mathscr{D} = C$.
- This is why, for two objects x and y, if $d_0(\Delta^k(x), \Delta^k(y)) \leqslant 1$ and $d_\infty(\Delta^k(x), \Delta^k(y)) \leqslant l$ for some k, then the same holds for all k', with $k' \leqslant k$.
- BOOL can be viewed as a divisive hierarchical clustering algorithm

Adjusted Rand Index

- Let the result be $\mathscr{C} = \{C_1, \ldots, C_K\}$ and the correct partition be $\mathscr{D} = \{D_1, \ldots, D_M\}$
- Suppose $n_{ij} := \|\{x \in X \mid x \in C_i, x \in D_j\}\|$. Then

$$\frac{\sum_{i,j} {}_{n_{ij}}C_2 - (\sum_i {}_{\|C_i\|}C_2 \sum_h {}_{\|D_j\|}C_2)/{}_nC_2}{2^{-1}(\sum_i {}_{\|C_i\|}C_2 + \sum_h {}_{\|D_j\|}C_2) - (\sum_i {}_{\|C_i\|}C_2 \sum_h {}_{\|D_j\|}C_2)/{}_nC_2}$$

Shape-Based Clustering Algorithms

- Many shape-based algorithms have been proposed
  - See [Berkhin, 2006; Halkidi *et al.*, 2001; Jain *et al.*, 1999]
- Partitional algorithms
  - ABACUS [Chaoji *et al.*, 2011], SPARCL [Chaoji *et al.*, 2009]
- Mass-based algorithms [Ting and Wells, 2010]
- Density-based algorithms
  - DBSCAN [Ester *et al.*, 1996]
- Hierarchical clustering algorithms
  - CURE [Guha *et al.*, 1998], CHAMELEON [Karypis *et al.*, 1999]
- Grid-based algorithms
  - STING [Wang *et al.*, 1997]

References

[Berkhin, 2006] P. Berkhin. A survey of clustering data mining techniques. *Grouping Multidimensional Data*, pages 25–71, 2006.

[Chaoji *et al.*, 2011] V. Chaoji, G. Li, H. Yildirim, and M. J. Zaki. ABACUS: Mining arbitrary shaped clusters from large datasets based on backbone identification. In *Proceedings of 2011 SIAM International Conference on Data Mining*, pages 295–306, 2011.

[Chaoji *et al.*, 2009] V. Chaoji, M. A. Hasan, S. Salem, and M. J. Zaki. SPARCL: An effective and efficient algorithm for mining arbitrary shape-based clusters. *Knowledge and Information Systems*, 21(2):201–229, 2009.

[Ester *et al.*, 1996] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of KDD*, 96, 226–231, 1996.

[Garcia-Molina *et al.*, 2008] H. Garcia-Molina, J. D. Ullman, and J. Widom. *Database systems: The complete book*. Prentice Hall Press, 2008.

[Guha *et al.*, 1998] S. Guha, R. Rastogi, and K. Shim. CURE: An efficient clustering algorithm for large databases. *Information Systems*, 26(1):35–58, 1998.

[Halkidi *et al.*, 2001] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, 2001.

[Hinneburg and Keim, 1998] A. Hinneburg and D. A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[Jain *et al.*, 1999] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[Karypis *et al.*, 1999] G. Karypis, H. Eui-Hong, and V. Kumar. CHAMELEON: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.

[Lin *et al.*, 2003] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 1–11, 2003.

[Qiu and Joe, 2006] W. Qiu and H. Joe. Generation of random clusters with specified degree of separation. *Journal of Classification*, 23:315–334, 2006.

[Sheikholeslami *et al.*, 1998] G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. In *Proceedings of the 24th International Conference on Very Large Data Bases*, pages 428–439, 1998.

[Ting and Wells, 2010] K. M. Ting and J. R. Wells. Multi-dimensional mass estimation and mass-based clustering. In *Proceedings of 10th IEEE International Conference on Data Mining*, pages 511 – 520, 2010.

[Wang *et al.*, 1997] W. Wang, J. Yang, and R. Muntz. STING: A statistical information grid approach to spatial data mining. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 186–195, 1997.