Lecture 9. Constraint-Based Clustering

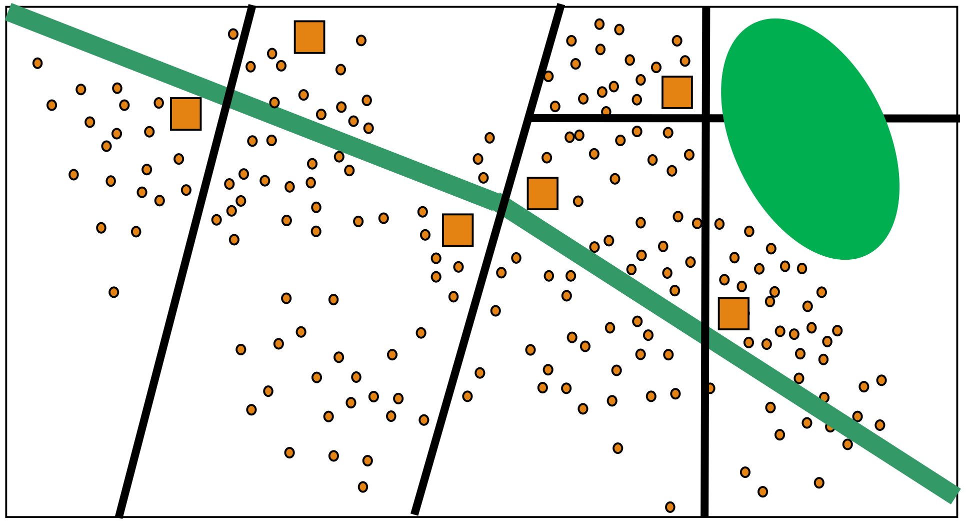
Lecture 9. Constraint-Based Clustering

- Why Constraint-Based Clustering?
- Categories of Constraints
- Constraint-Based Clustering: Handling Hard Constraints
- Constraint-Based Clustering: Handling Soft Constraints
- Constraint-Based Clustering: Constraints on Distance Measures
- User-Guided Clustering: Taking User's *Hints* as Constraints
- Summary

Session 1. Why Constraint-Based Clustering?

Why Constraint-Based Cluster Analysis?

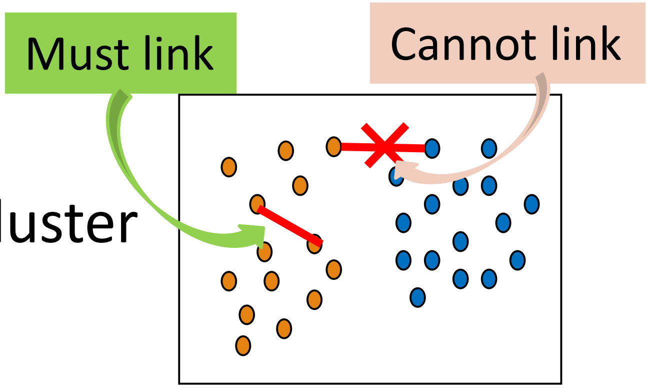
- Constraint-based clustering: Clustering with user-specified constraints
  - Semi-supervised clustering: Often in the form of “cannot link” and “must link”
  - Constraint-based clustering can be much broader, e.g., distance constraints, user-guidance, user-specified # of clusters, granularity of clusters, etc.
- Constraint-based clustering is highly desirable to guide clustering
  - Users know their applications the best
  - Less parameters but more user-desired constraints
- Application examples
  - Clustering students: for awards or for parties?
  - Allocating delivery centers? Need to consider obstacles (highways, rivers, lakes and mountains), available roads, traffic, etc.



Session 2. Categories of Constraints

Categorization of Constraints

- Constraints **on instances**: Specify how a pair or a set of instances should be grouped in the cluster analysis
  - **Must-link** vs. **cannot-link** constraints
    - *must-link*(*x, y*): Objects *x* and *y* should be grouped into one cluster
  - Constraints can be defined using variables
    - Ex. *cannot-link*(*x, y*) if *dist*(*x, y*) > *d*

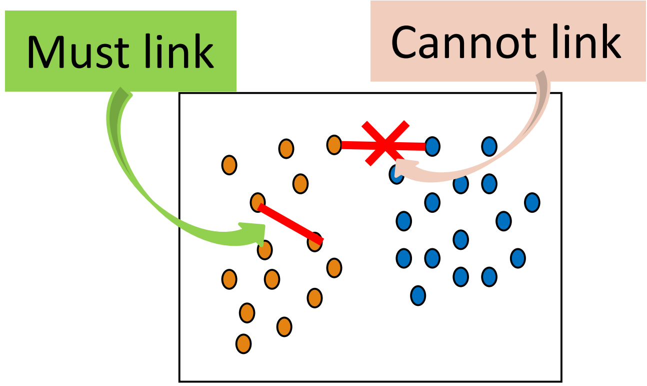


- Constraints **on clusters**: Specify a requirement of the clusters
  - Ex. Specify the minimum number of objects in a cluster, the maximum diameter of a cluster, the shape of a cluster (e.g., a convex), # of clusters (e.g., *k*)
- Constraints **on distance measures**
  - Specify a requirement that the distance calculation must respect
  - Ex. Driving on roads, observing obstacles (e.g., rivers, lakes)
- Issues: Hard vs. soft constraints; conflicting or redundant constraints

Session 3. Constraint-Based Clustering: Handling Hard Constraints

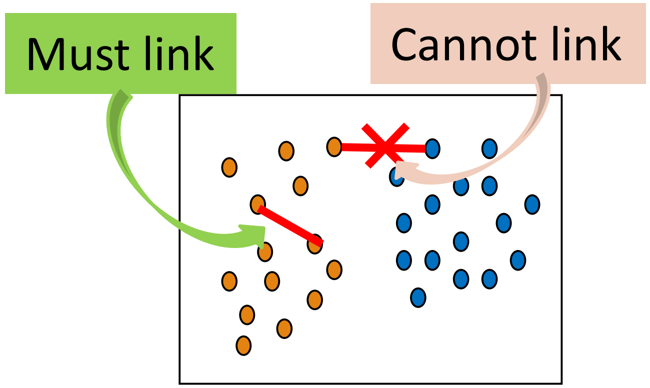
Constraint-Based Clustering: Handling Hard Constraints

- Handling hard constraints: **Strictly respect the constraints in cluster assignments**
- Example: The **COP-*k-means*** algorithm
  - **Generate super-instances for must-link constraints**
    - Compute the transitive closure of the must-link constraints
    - To represent such a subset, replace all those objects in the subset by the mean
    - The super-instance also carries a weight, which is the number of objects it represents
  - **Conduct modified *k-means* clustering to respect cannot-link constraints**
    - Modify the center-assignment process in *k-means* to a *nearest feasible center assignment*
    - An object is assigned to the nearest center so that the assignment respects all cannot-link constraints



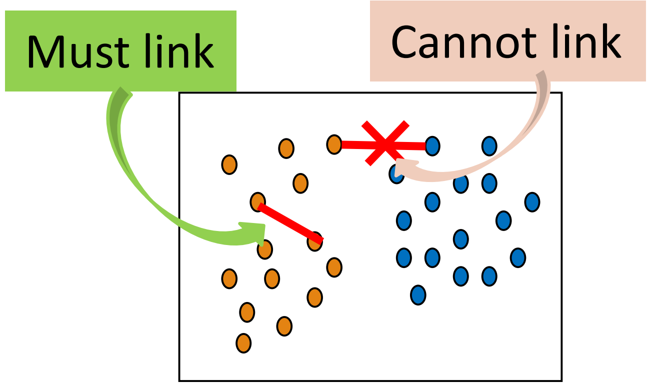
Session 4. Constraint-Based Clustering: Handling Soft Constraints

Constraint-Based Clustering: Handling Soft Constraints



- Treated as an **optimization problem**
  - When a clustering violates a soft constraint,
    a penalty is imposed on the clustering
- Overall objective
  - Optimizing the clustering quality and minimizing the constraint violation penalty

Handling Soft Constraints: An Example Algorithm



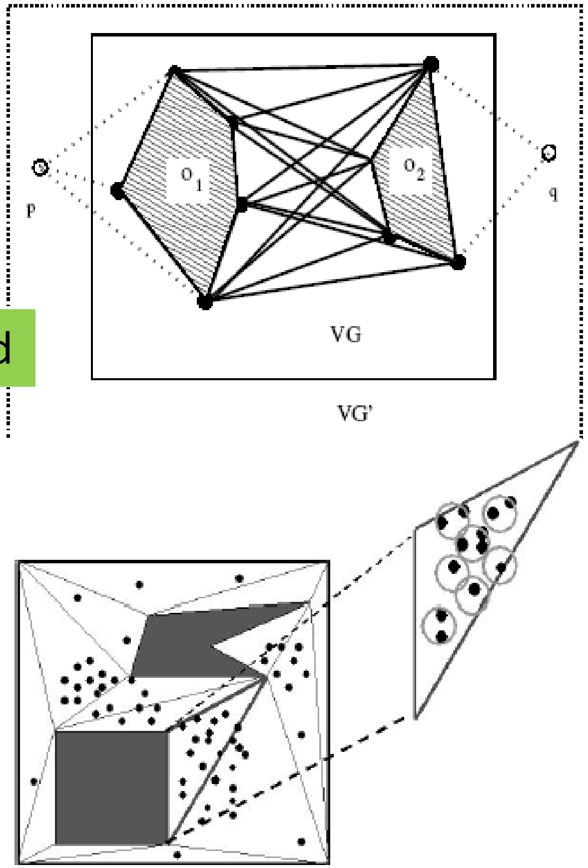
- CVQE (Constrained Vector Quantization Error) algorithm
  - Conduct *k-means* clustering while enforcing constraint violation penalties
- Objective function
  - Sum of distance used in *k-means*, adjusted by the constraint violation penalties
  - Penalty of a *must-link* violation
    - If objects *x* and *y* must-be-linked but they are assigned to two different centers, c_1 and c_2, *dist*(c_1, c_2) is added to the objective function as the penalty
  - Penalty of a *cannot-link* violation
    - If objects *x* and *y* cannot-be-linked but they are assigned to a common center *c*, *dist*(*c*, *c′*), between *c* and *c′* is added to the objective function as the penalty, where *c′* is the closest cluster to *c* that can accommodate *x* or *y*

Session 5. Constraint-Based Clustering: Constraints on Distance Measures

Constraints on Distance Measures: Efficient Processing

- Spatial clustering in the presence of obstacles (Tung, et al., ICDE'01)
- It is preferable to use *K-medoids*
  - *K-means* may locate the service center in the middle of a lake

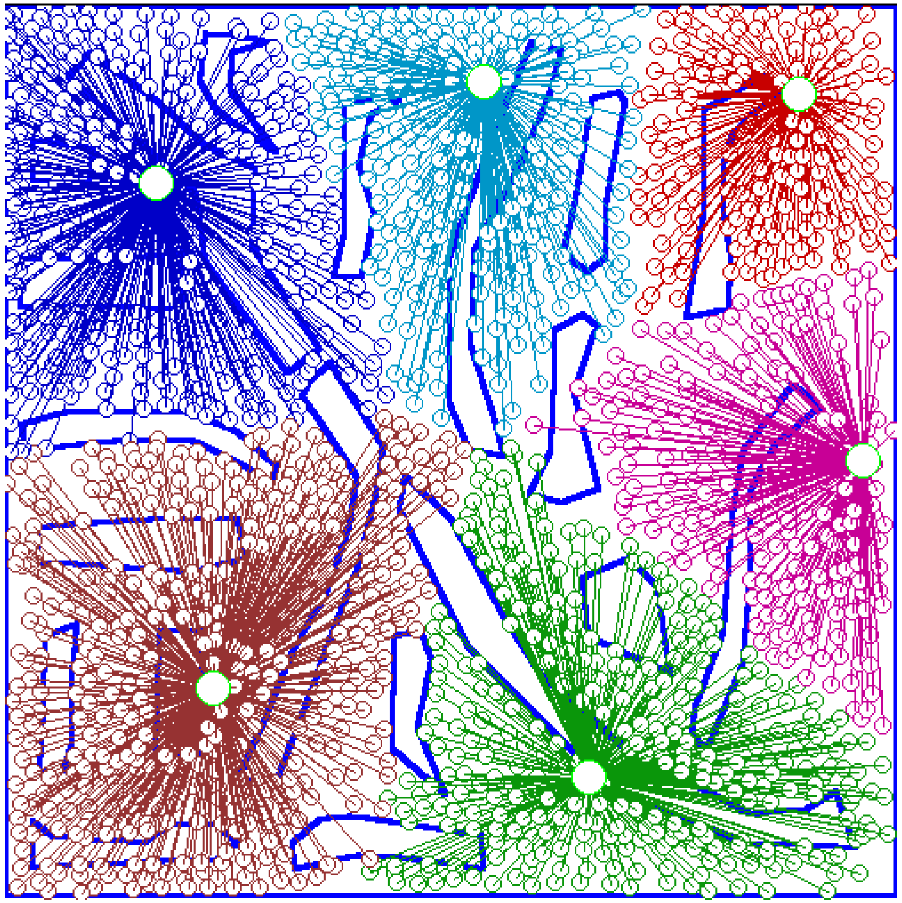
Obstacle-observed clustering is needed



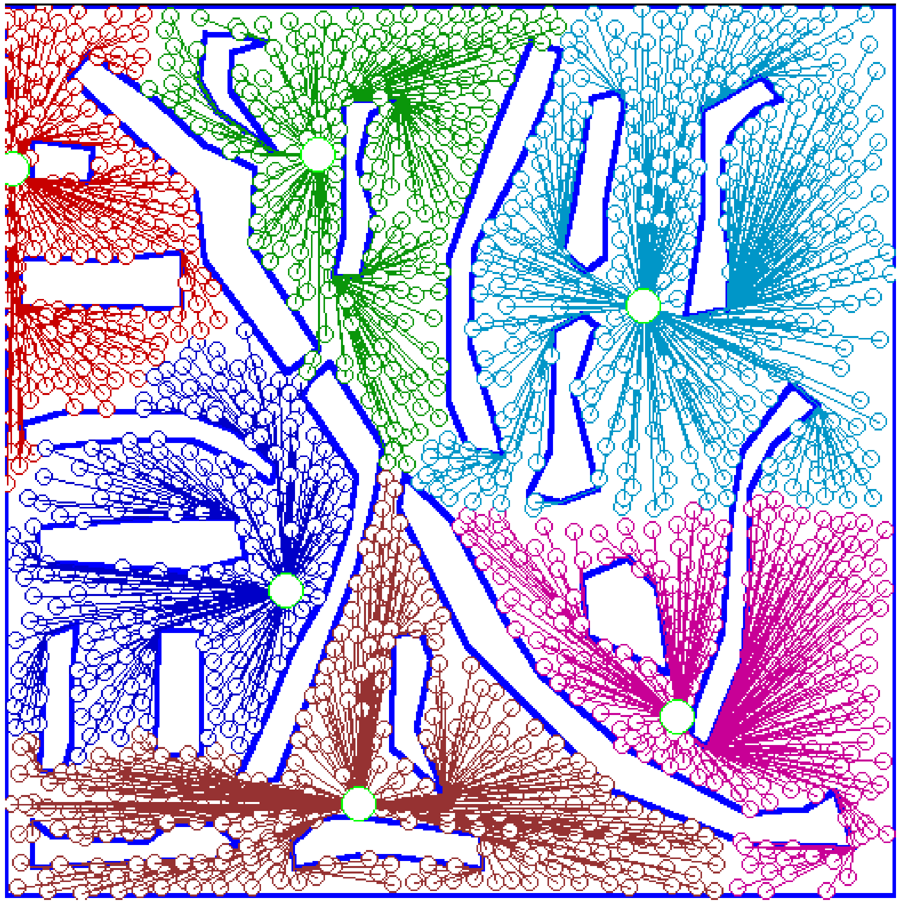
- It is costly to compute such constrained clustering
  - Re-compute distance between a point with new centroids
- Need to compute the visibility graph and shortest paths
  - Triangulation and micro-clustering
- Two kinds of join indices (shortest-paths) worth pre-computation
  - VV index: Indices for any pair of obstacle vertices
  - MV index: Indices for any pair of micro-cluster and obstacle indices

An Example: Clustering With Obstacle Objects

- Compare clustering results: Without the obstacle constraints vs. with the constraints



Without enforcing the obstacle constraint

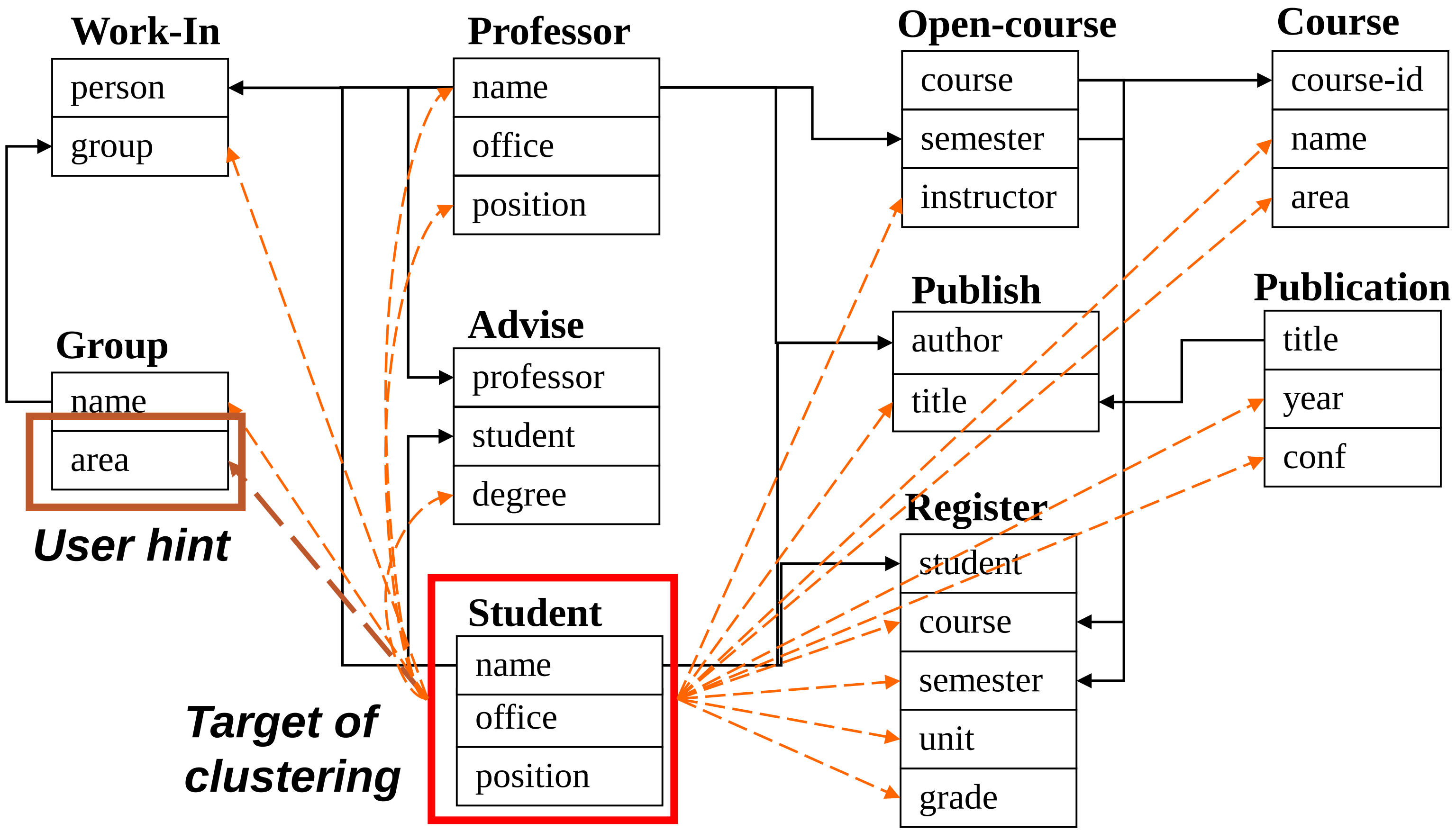


Enforcing the obstacle constraint

Session 6. User-Guided Clustering: Taking User's *Hints* as Constraints

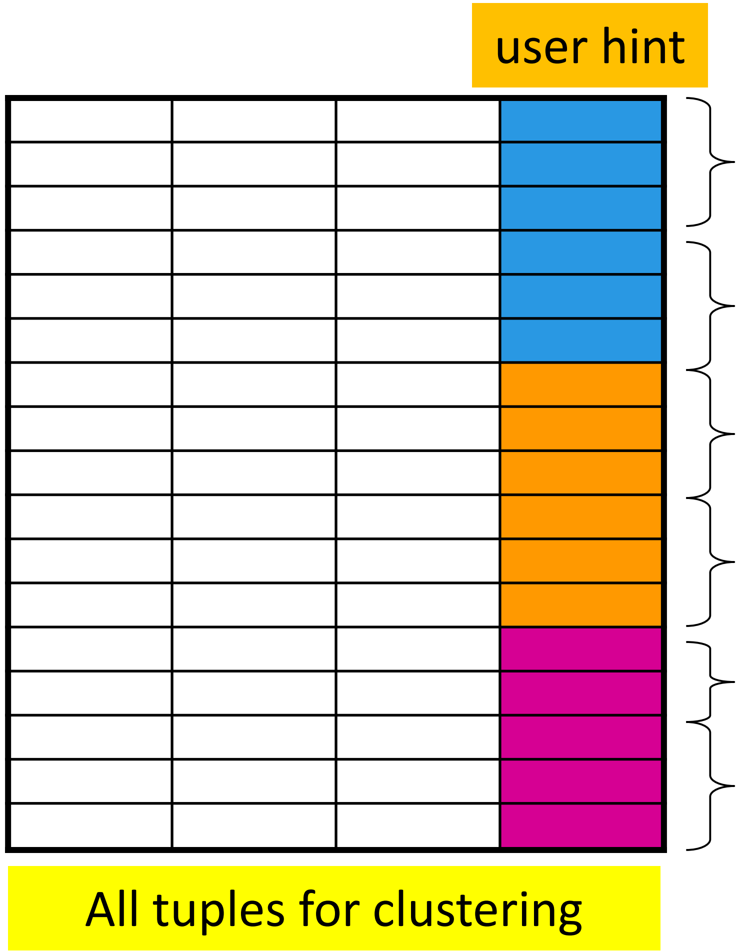
User-Guided Clustering: A Special Kind of Constraints

- X. Yin, J. Han, P. S. Yu, "Cross-Relational Clustering with User's Guidance", KDD'05
- The goal or purpose of a clustering task should be specified by users
- Example: Clustering students by research area
- "Students" are linked with many other relations in a database



- It is not easy for a user to provide a good training set or a set of clear constraints
- It is much easier for a user to specify an attribute as a ***hint***, such as the *area* (or *field*) *of a* student's *research group* for ***user-guided clustering***

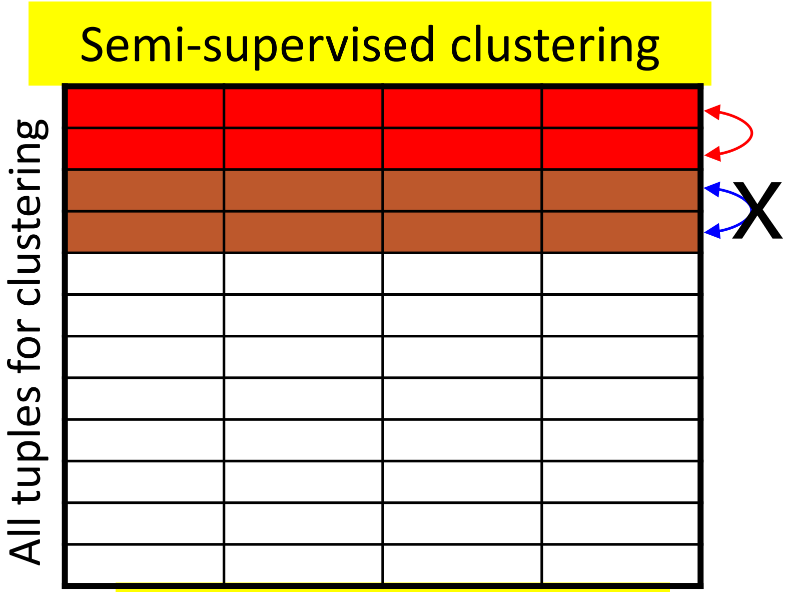
User-Guided Clustering Is Different from Classification

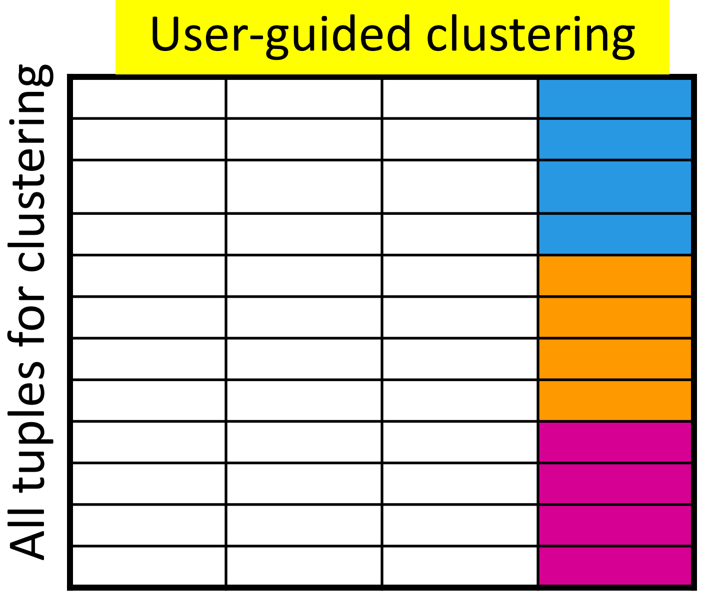


- User-specified *feature* (in the form of *attribute*) is used as a hint, not class labels
  - The attribute may contain too many or too few distinct values
    - E.g., a user may want to cluster students into 20 clusters instead of 3
  - Additional features need to be included in cluster analysis across multiple relations

User-Guided Clustering Is Different from Semi-Supervised Clustering

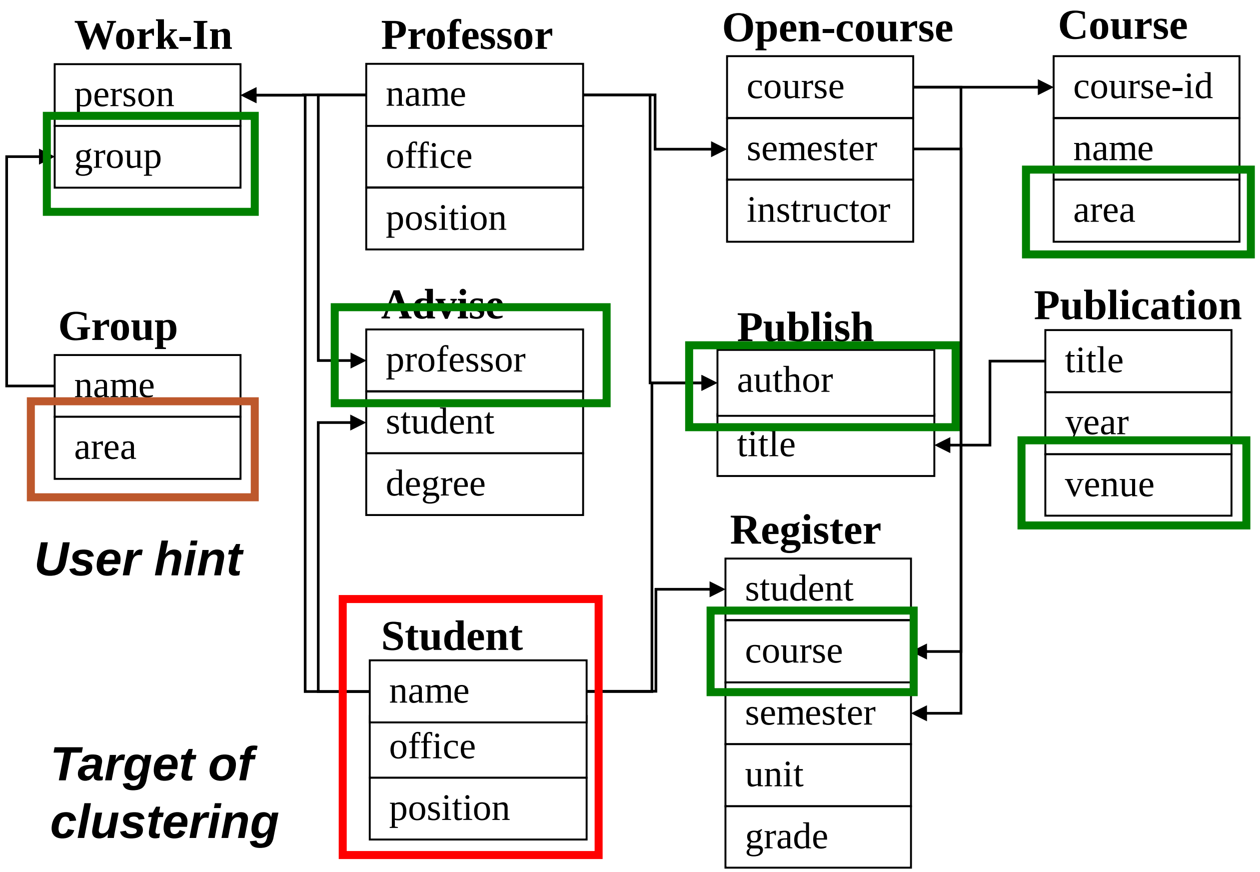
- **Semi-supervised clustering**: User provides a training set consisting of *similar* (*must-link*) and *dissimilar* (*cannot link*) pairs of objects
- **User-guided clustering**: User specifies an attribute as a hint, and more relevant features are found for clustering
- **Why not semi-supervised clustering?**
  - Much information (in multiple relations) is needed to judge whether two tuples are similar
  - A user may not be able to specify a set of good constraints but it is easy to provide hints





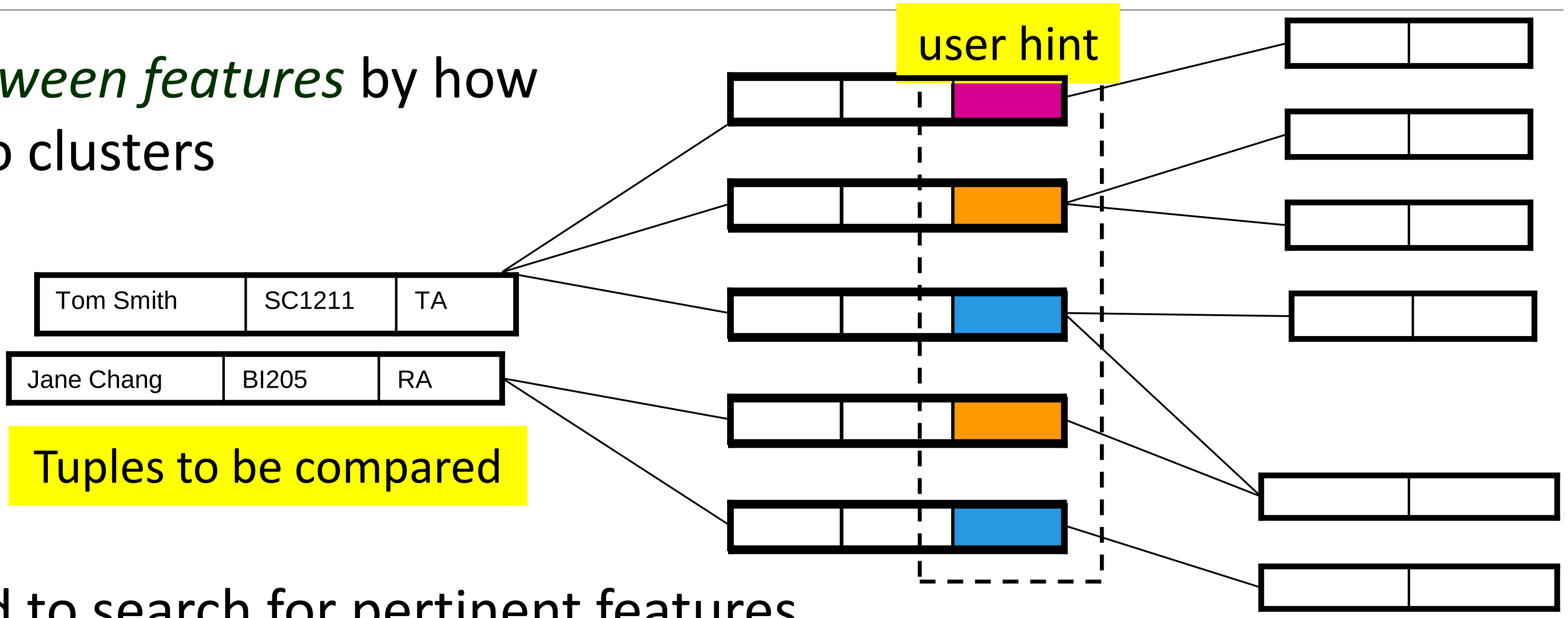
Heuristic Search for Pertinent Features

- Overall procedure
  - Start from the user-specified feature
  - Search in neighborhood of existing pertinent features
  - Expand search range gradually
- Tuple ID propagation is used to create multi-relational features
  - IDs of target tuples can be propagated along any join path, from which we can find tuples joinable with each target tuple



CrossClus: User-Guided Clustering Cross Multiple Relations

- ❑ Measure *similarity between features* by how they group objects into clusters



- ❑ Use a heuristic method to search for pertinent features
  - ❑ *Start from user-specified feature and gradually expand search range*
- ❑ Use *tuple ID propagation* to create feature values
  - ❑ Features can be easily created during the expansion of search range by propagating IDs
- ❑ Explore three clustering algorithms: *k-means*, *k-medoids*, and hierarchical clustering

Finding Features Across Multiple Relations

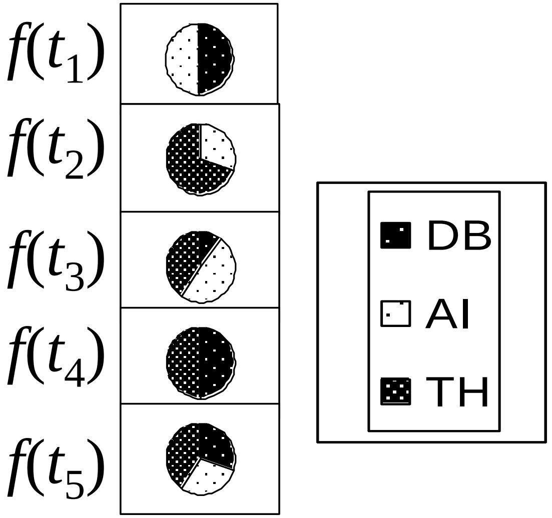
- Measure *similarity between features* by how they group objects into clusters
- A multi-relational feature is defined by:
  - A join path, e.g., *Student → Register → OpenCourse → Course*
  - An attribute, e.g., *Course.area*
  - (For numerical feature) an aggregation operator, e.g., sum or average
- Categorical feature *f* = [*Student → Register → OpenCourse → Course, Course.area*]

areas of courses of each student

Tuple	Areas of courses		
	DB	*AI*	*TH*
t_1	5	5	0
t_2	0	3	7
t_3	1	5	4
t_4	5	0	5
t_5	3	3	4

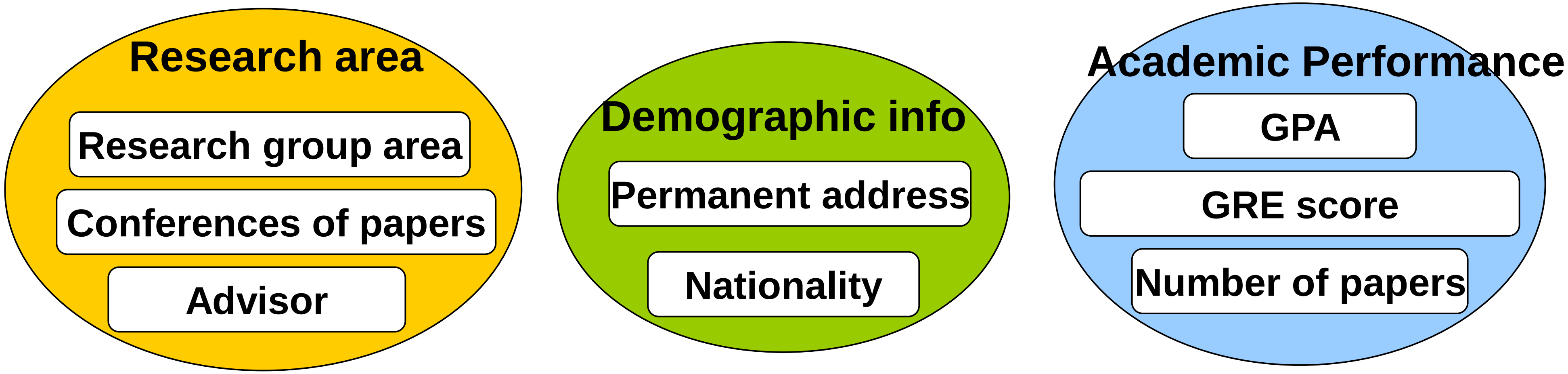
values of feature *f*

Tuple	Feature *f*		
	DB	*AI*	*TH*
t_1	0.5	0.5	0
t_2	0	0.3	0.7
t_3	0.1	0.5	0.4
t_4	0.5	0	0.5
t_5	0.3	0.3	0.4



Searching for Pertinent Features

- Different features convey different aspects of information



- Features conveying same aspect of information usually cluster tuples in more similar ways
  - Research group areas vs. conferences of publications
- Given user-specified feature, find pertinent features by computing feature similarity

Clustering with Multi-Relational Features

- Given a set of L pertinent features f_1, …, f_L, similarity between two tuples

$$\mathrm{sim}(t_1, t_2) = \sum_{i=1}^{L} \mathrm{sim}_{f_i}(t_1, t_2) \cdot f_i.weight$$

  - Weight of a feature is determined in feature search by its similarity with other pertinent features
- Experimented on multiple clustering methods, including
  - CLARANS, *K*-means, and agglomerative hierarchical clustering
- Compared with a subspace clustering algorithm, PROCLUS, and an ILP clustering algorithm, RDBC [Kirsten and Wrobel'00]
  - The performance shows that user-guided clustering, CrossClus, leads to better clustering accuracy—validated using given labels on real datasets (ground truth)

Session 8: Summary

Summary: Constraint-Based Clustering

- Why Constraint-Based Clustering?
- Categories of Constraints
- Constraint-Based Clustering: Handling Hard Constraints
- Constraint-Based Clustering: Handling Soft Constraints
- Constraint-Based Clustering: Constraints on Distance Measures
- User-Guided Clustering: Taking User's "Hints" as Constraints
- Summary

Recommended Readings

- A. K. H. Tung, J. Hou, and J. Han. Spatial Clustering in the Presence of Obstacles. ICDE’01
- A. K. H. Tung, J. Han, L. V. S. Lakshmanan, and R. T. Ng. Constraint-Based Clustering in Large Databases. ICDT’01
- I. Davidson and S. S. Ravi. Clustering with Constraints: Feasibility Issues and the K-Means Algorithm. SDM’05
- X. Yin, J. Han, and P. S. Yu. Cross-Relational Clustering with User's Guidance. KDD'05
- I. Davidson, K. L. Wagstaff, and S. Basu. Measuring Constraint-Set Utility for Partitional Clustering Algorithms. PKDD’06
- J. Han, M. Kamber, and J. Pei. Data Mining: Concepts and Techniques. Morgan Kaufmann, 3rd ed. , 2011
- A. Agovic and A. Banerjee, Semi-Supervised Clustering, in (Chapter 20) C. Aggarwal and C. K. Reddy (eds.), Data Clustering: Algorithms and Applications. CRC Press, 2014
- C. Aggarwal. Data Mining: The Textbook. Springer, 2015