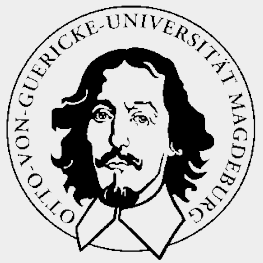
Data & Knowledge Engineering Group

Everything in its right place?

Learning a user's view of a music collection

March 24, 2009 @ NAG / DAGA'09

Korinna Bade, Andreas Nürnberger & Sebastian Stober
Otto-von-Guericke-University Magdeburg
http://www.dke-research.de



Outline

- Motivation
- Hierarchical Agglomerative Clustering
- Defining Constraints for Hierarchies
- Method 1:
  - Instance-based Constrained HAC
- Method 2:
  - Metric Learning for HAC
- Comparison
- Discussion:
  - Generalizations and Limitations

Motivation

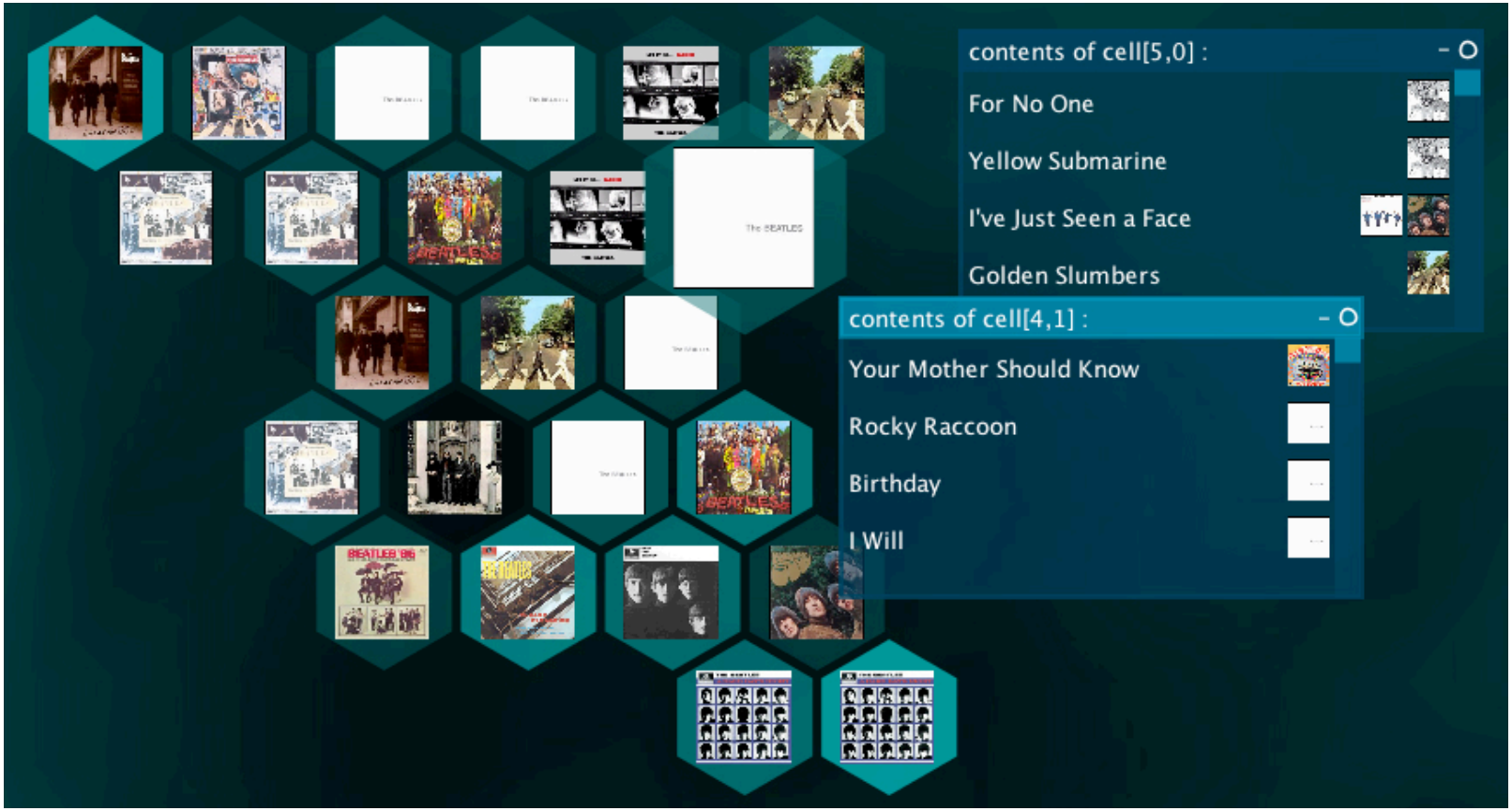
- support music collection browsing & exploration
- increase awareness and accessibility of data

goal: intuitively understandable structuring through personalization

i.e. don't force the user to learn how the system works – let the system learn what the user wants

So far (DAGA'08)

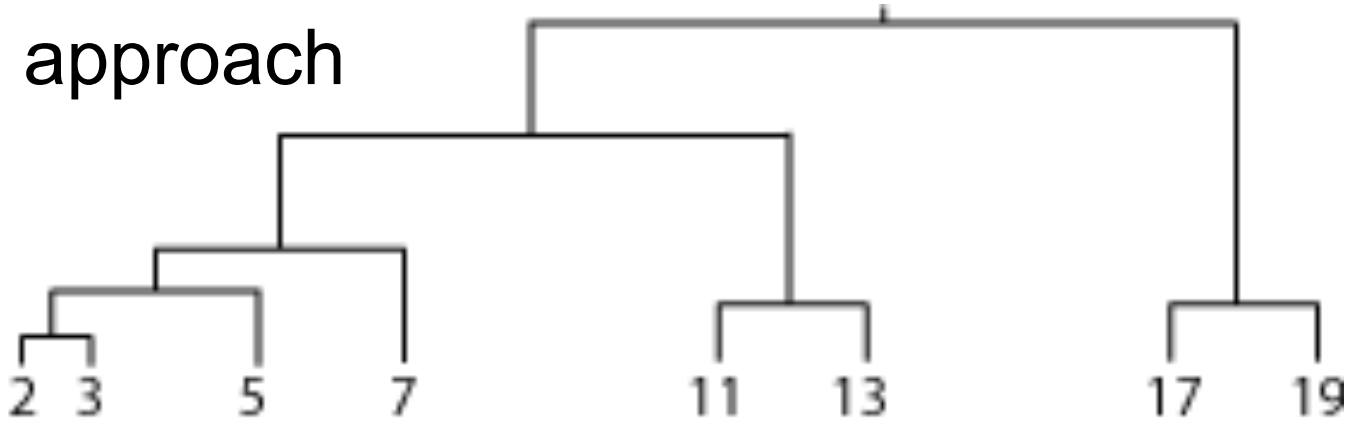
- adaptive Self-Organizing Map:



- limitation: flat clustering does not scale well

Hierarchical Agglomerative Clustering

- builds a hierarchy of partitions of the data visualized as a dendrogram
- bottom-up approach



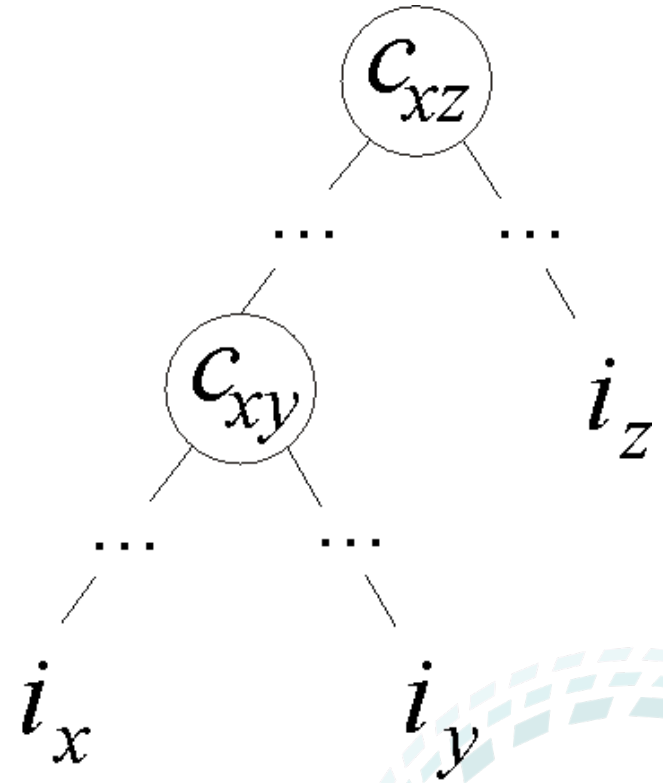
algorithm:

1. start with each item as a separate cluster
2. repeat until only a single cluster is left:
   1. merge the two most similar clusters (from the current top-most partition)
   2. add the new partition on top of the hierarchy

Defining Constraints for Hierarchies

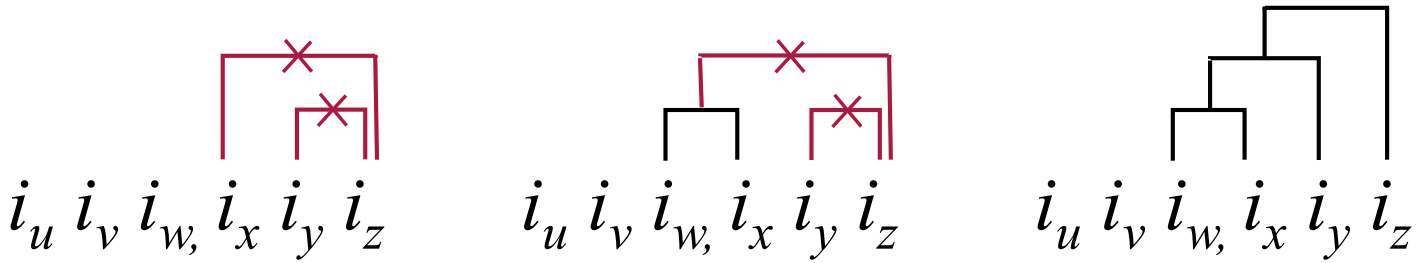
- Must-link-before (MLB) constraints
  - ordering of item linkage
  - triple (i_x, i_y, i_z) means: items i_x and i_y should be linked on a lower level than i_x and i_z

- MLB properties:
  - symmetry: $(i_x, i_y, i_z) \rightarrow (i_y, i_x, i_z)$
  - transitivity:
    - $(i_x, i_y, i_z) \wedge (i_y, i_w, i_z) \rightarrow (i_x, i_w, i_z)$
    - $(i_x, i_y, i_z) \wedge (i_y, i_z, i_w) \rightarrow (i_x, i_y, i_w)$



Instance-based Adaptation

- merge clusters only in accordance to constraints
- for all MLB constraints (i_x, i_y, i_z):
  the cluster containing i_z can only be merged with a cluster containing both, i_x and i_y, or neither



- if no merge is possible due to constraints
  - stop early or
  - merge clusters violating the fewest constraints

Method 2: Metric Learning

- idea: interpret a constraint as a relation between item similarities: $(i_x, i_y, i_z) \Rightarrow \mathrm{sim}(i_x, i_y) > \mathrm{sim}(i_x, i_z)$
- parameterize similarity measure (weighting of the different dimensions)
- learn a weighting scheme that leads to a hierarchy violating as few constraints as possible

gradient descend:

1. initialize weights with 1 (standard similarity)
2. repeat until convergence:
   1. for each violated constraint adapt weights: $w_j \leftarrow w_j + \eta \frac{\partial\left(\mathrm{sim}(i_x, i_y) - \mathrm{sim}(i_x, i_z)\right)}{\partial w_j}$

      (i.e. make i_x and i_y more similar and i_x and i_z more dissimilar)

Comparison

- instance-based adaptation:
  - rather local effects
  - sufficient number of constraints needed for good generalization
  - no requirements on item representation
  - “hard-wired” items → need to be in the collection
- metric learning:
  - more global influence
    → change in the underlying similarity measure
  - similar items should be handled (almost) equally
  - requires parameterized similarity measure
  - can use independent training set of constraints
  - can be applied to any other (unknown) collection
- can nicely be combined

Discussion

- adaptation approaches can be applied to other hierarchical clustering methods as well
  - e.g. Bisecting k-Means, GHSOM
- works with any parameterized differentiable similarity measure
  - usually weighted sum of feature similarities
- Problem: What if user wants to assign an item to two distinct places in a hierarchy?
  a) considers different aspects

     → support different (parallel) views on the collection
  b) two distinct characteristics for a single aspect

Thank You for Your Attention!

QUESTIONS…?

ongoing survey at http://survey.dke-research.de
“How much may your music player know about your listening habits?”