High-Order Heterogeneous Data Mining

Tie-Yan Liu
Researcher, Microsoft Research Asia
2006-11-5
Why High-Order Heterogeneous?

• The world is heterogeneous
  – Objects are heterogeneous:
    • (query, document...), (author, paper...)

• Many applications involve multiple types of objects
  – Web search
    • User ↔ Query ↔ Web Page
  – Academic society
    • Author ↔ Paper ↔ Conference
      ↑
      ↓
      Journal
  – Relationships among these objects are also heterogeneous:
    similarity, relevance, endorsement; directed, undirected...
However, ...

- Most traditional ML and DM methods focus on homogeneous data, or data of no more than two types.
Related Work: Spectral Clustering

(PAMI 2000)

- Spectral clustering cuts relationship graph to cluster similar data.
- Minimize graph cut

\[
obj = \frac{\text{cut}(V_1, V_2)}{\text{weight}(V_1)} + \frac{\text{cut}(V_2, V_1)}{\text{weight}(V_2)}
\]

\[
\text{cut}(V_1, V_2) = \sum_{i \in V_1, j \in V_2, i < j} e_{ij}
\]

and

\[
\text{weight}(V_i) = \sum_{j \in V_i} W_j.
\]

\[
\min \frac{q^T L q}{q^T D q}, \text{ subject to } q^T D e = 0, q \neq 0
\]

- Solution
  - Graph cut can be converted to a generalized eigenvalue problem by using continuous slacking: \( Lq = \lambda Dq \)
  - The eigenvector associated with the second smallest eigenvalue of the Laplace matrix is an optimal embedding for cut minimization.
Related Work: PageRank
(WWW 1998)

• PageRank ranks the popularity of vertices in a directed graph according to their linkage information.

• PageRank of a vertex is proportional to its parents’ rank, but inversely proportional to its parents’ outdegree.

\[ R(u) = d + (1 - d) \sum_{v \in B_u} \frac{R(v)}{N_v} \]

\[ R = (1 - d)AR + d\Pi, \quad A_{u,v} = \frac{1}{N_v}, \quad \Pi = \frac{1}{N} [1,1,\ldots,1] \]

• PageRank can be explained using a Markov random surfer model; or be explained as the principal eigenvector of the smoothed adjacency matrix of the Web graph.
Related Work: Bipartite Graph Partitioning  
(KDD 2001)

- Cuts bipartite relationship graph to cluster two types of data simultaneously.

\[ M = X \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \]

- Due to the bipartite property of the graph, after some trivial deduction, this problem can be converted to a singular value decomposition (SVD) problem.
Related Work: Information Theoretic Co-Clustering  
(KDD 2003)

\[ C_X : \{x_1, ..., x_m\} \rightarrow \{\hat{x}_1, ..., \hat{x}_r\} \]
\[ C_Y : \{y_1, ..., y_n\} \rightarrow \{\hat{y}_1, ..., \hat{y}_s\} \]

- An optimal co-clustering minimizes \( I(X, Y) - I(\hat{X}, \hat{Y}) \)
subject to the constraints on the number of row and column clusters.

It can be proved that
\[
I(X, Y) - I(\hat{X}, \hat{Y}) = D(p(X, Y) \parallel q(X, Y))
\]
where \( D(,) \) denotes the KL divergence, and
\( q(X,Y) \) is a distribution of the form
\[
q(x, y) = p(\hat{x}, \hat{y}) p(x \mid \hat{x}) p(y \mid \hat{y})
\]

[Step 1] Set \( i = 1 \). Start with \((R_i, C_i)\), Compute \( q[i,i] \).

[Step 2] For every row \( x \), assign it to the cluster \( \hat{x} \) that minimizes
\[
KL(p(y \mid x) \parallel q[i,i](y \mid \hat{x}))
\]

[Step 3] We have \((R_{i+1}, C_i)\). Compute \( q[i+1,i] \).

[Step 4] For every column \( y \), assign it to the cluster \( \hat{y} \) that minimizes
\[
KL(p(x \mid y) \parallel q[i+1,i](x \mid \hat{y}))
\]

[Step 5] We have \((R_{i+1}, C_{i+1})\). Compute \( q[i+1,i+1] \). Iterate 2-5.
Going Beyond...

• Modeling the relationships
  – Unified Relationship Matrix
  – Tensor
  – Collective bipartite graphs

• Designing effective data mining algorithms
  – High-order Heterogeneous Coclustering
  – High-order Heterogeneous Coranking
Unified Relationship Matrix

• Integrate pairwise relationship matrices into a unified matrix

\[
L = \begin{bmatrix}
\lambda_{11}L_1 & \lambda_{12}L_{12} & \cdots & \lambda_{1N}L_{1N} \\
\lambda_{21}L_{21} & \lambda_{22}L_2 & \cdots & \lambda_{2N}L_{2N} \\
\vdots & \vdots & \ddots & \vdots \\
\lambda_{N1}L_{N1} & \lambda_{N2}L_{N2} & \cdots & \lambda_{NN}L_N \\
\end{bmatrix}
\]

\[L_{urm} = D^{-1}L\]

Combination coefficients can be manually set or learned from labeled data

• Representative Work
  – Wensi Xi, et al, SimFusion: Measuring Similarity using Unified Relationship Matrix, **SIGIR 2005**.
  – Xuanhui Wang, et al, Latent Semantic Analysis for Multiple-Type Interrelated Data Objects, **SIGIR 2006**.
Tensor

- Use multi-linear algebra to represent heterogeneous relationship.

- Representative Work
Collective Bi-partite Graphs

• Decompose heterogeneous relationship into a collections of pairwise relationships.

• Representative Work
  – Bin Gao, Tie-Yan Liu, et al, Consistent Bipartite Graph Co-Partitioning for Star-Structured High-Order Heterogeneous Data Co-Clustering, *KDD 2005*.
Algorithms

• Unified Relationship Matrix
  – LinkFusion (WWW 2004)
  – Object-level Ranking (WWW 2005)
  – SimFusion (SIGIR 2005)
  – Multi-type LSA (SIGIR 2006)

• Tensor
  – CubeSVD (WWW 2005)

• Collective Bipartite Graphs
  – Consistent Bipartite Graph Co-partitioning (KDD 2005)
  – Consistent Information-theoretic Co-clustering (ICDM 2006)
  – Generalized SVD for Co-clustering (IEEE TKDE)
  – Spectral Clustering for Multi-type Relational Data (ICML 2006)
Link Fusion

- High-order heterogeneous version of PageRank
Random Walk on Heterogeneous Graph

- Construct the URM by merging pair-wise PageRank matrices with manually-set combination coefficients.
- Imagine a Markov random walk over the heterogeneous graph represented by the URM.
- Ranking over heterogeneous data will correspond to the principle eigenvector of the URM: \( w = L_{urm}^T w \), and the convergence can be proven.
Object-Level Ranking

- Use similar URM formulation to LinkFusion
- Learn the combination coefficients with a training set.

\[ R_X = \varepsilon R_{EX} + (1 - \varepsilon) \sum_{Y} \gamma_{YX} M_{YX}^T R_Y \]
Learning the Coefficients

Subgraph Selection
Starting with the labeling data objects, and including all other objects with less than $k$-step links from them.

Parameter Search
Using simulated annealing based method to search the best parameter in the selected subgraph.

---

**Algorithm DiameterEstimator($\delta$: stopping threshold)**

```
for (each object type $X$)
    $n \leftarrow$ total number of different object types related to objects of type $X$;
    for (each related object type $Y$) $\gamma_{YX} \leftarrow \frac{1}{n}$;
end for
compute the PopRank scores over the entire graph;
$R' \leftarrow$ the ranking vector of the training objects;
$k \leftarrow 0$;
while($||R - R'||_1 > \delta$) $k++$;
    compute the PopRank scores over the $k$ diameter subgraph;
    $R' \leftarrow$ the ranking vector of the training objects;
end while
return $k$;
```

---

**Algorithm SAFA (timeout: stopping condition)**

```
for (each object type $X$)
    $n \leftarrow$ total number of different object types related to objects of type $X$;
    for (each related object type $Y$) $\gamma_{YX} \leftarrow \frac{1}{n}$;
end for
$t \leftarrow$ a large number;
do
    for (each object type $X$)
        for (each object type $Y$)
            repeat
                repeat
                    randomly select $\gamma_{YX}$ in $\text{Neighbor}(\gamma_{YX})$;
                    $\text{diff} \leftarrow f(\gamma_{YX}) - f(\gamma_{YX})$;
                    if $\text{diff} > 0$ then $\gamma_{YX} \leftarrow \gamma_{YX}$;
                    else generate random $x$ in $(0,1)$;
                    if $x < \exp(-\text{diff}/t)$ then $\gamma_{YX} \leftarrow \gamma_{YX}$;
                until iteration count $= max\_number\_iteration$;
            $t \leftarrow 0.9t$;
            until iteration count $= max\_number\_iteration$;
        end for
    end for
until timeout;
return the best combination of $\gamma_{YX}$;
```

---
SimFusion

The similarity of two data objects in one data type can be reinforced by the similarity value of other data objects they are related to.
Mathematical Formulation

• The similarity reinforcement assumption can be represented as:
  
  - \[ S^{\text{new}} = L_{urm} S^{\text{original}} L_{urm}^T \]
  
  - \[ S^n = L_{urm} S^{n-1} L_{urm}^T = L_{urm}^n S^0 (L_{urm}^n)^T \]
  
  - Convergence can be proven.

• The so-calculated similarity can be used for many applications such as object clustering and information retrieval.
Multi-type LSA

• The Mutual Reinforcement Principle of LSA
  – On a multiple-type graph \( G \) with \( N \) vertices and a number of pairwise co-occurrence relationships, \textit{important} objects of a type co-occur with \textit{important objects} of other types.
Low Rank Approximation

• Conduct EVD on the URM
• Apply similar ideas to principal component analysis, we can regard top k eigenvectors as representing the top k important concepts, and use them to span a k-dimensional semantic space to represent all the objects.
• Use the low-rank approximation of the URM to capture latent semantics, just as classical LSA does.
Discussions on URM

- **Pros**
  - By building URM, traditional methods for homogeneous data can be easily used.
  - Linear algebra might be the most mature mathematical tool in data mining.

- **Cons**
  - Basic assumption in these approaches is questionable: is it really reasonable that heterogeneous relationship can become homogeneous with linear scaling?
Algorithms

- Use ??????????????????"??????????????????????????????????????????????
CubeSVD

- Matrix Singular Value Decomposition (SVD)
  - Latent Semantic Indexing (LSI)
    - Apply SVD on document-term matrix
  - In Recommender System
    - Apply SVD on user-item preference matrix
CubeSVD (cont.)

• Tensor Singular Value Decomposition (High-order SVD)
  – Higher-Order SVD might also capture the latent factors that govern the relations among multi-type objects.
  – These semantic relationships can be used to get better clustering.

\[
\mathcal{A} = S \times_1 V_1 \times_2 V_2 \cdots \times_N V_N
\]

1. Construct tensor \( \mathcal{A} \) from the clickthrough data. Suppose the numbers of user, query and Web page are \( m, n, k \) respectively, then \( \mathcal{A} \in \mathbb{R}^{m \times n \times k} \). Each tensor element measures the preference of a \( \text{(user, query)} \) pair on a Web page.
2. Calculate the matrix unfolding \( A_u, A_q \) and \( A_p \) from tensor \( \mathcal{A} \). \( A_u \) is calculated by varying user index of tensor \( \mathcal{A} \) while keeping query and page index fixed. \( A_q \) and \( A_p \) are computed in a similar way. Thus \( A_u, A_q, A_p \) is a matrix of \( m \times nk, n \times mk, k \times mn \) respectively.
3. Compute SVD on \( A_u, A_q \) and \( A_p \), set \( V_u, V_q \) and \( V_p \) to be the left matrix of the SVD respectively.
4. Select \( m_0 \in [1, m], n_0 \in [1, n] \) and \( k_0 \in [1, k] \). Remove the right-most \( m - m_0, n - n_0 \) and \( k - k_0 \) columns from \( V_u, V_q \) and \( V_p \), then denote the reduced left matrix by \( W_u, W_q \) and \( W_p \) respectively. Calculate the core tensor as follows:
\[
S = A \times_1 W_u^T \times_2 W_q^T \times_3 W_p^T
\]
5. Reconstruct the original tensor by:
\[
\hat{\mathcal{A}} = S \times_1 V_u \times_2 V_q \times_3 V_p
\]
Discussions on Tensor

• Pros
  – Using tensor to represent heterogeneous data objects is more natural than URM

• Cons
  – Multi-linear algebra is in its initial stage, and many basic operations for tensor have not been reasonably define.
    • Tensors cannot always be “diagonalized”
    • $k$ successive rank-1 approximations to tensors do not necessarily result in the best rank-$k$ approximation
    • Eight factors about tensor
  – Complexity of tensor operator is very high, thus tensor based methods are difficult to scale up.
Algorithms

• Unified Relationship Matrix
  – LinkFusion (WWW 2004)
  – Object-level Ranking (WWW 2005)
  – SimFusion (SIGIR 2005)
  – Multi-type LSA (SIGIR 2006)

• Tensor
  – CubeSVD (WWW 2005)

• Collective Bipartite Graphs
  – Consistent Bipartite Graph Co-partitioning (KDD 2005)
  – Consistent Information-theoretic Coclustering (ICDM 2006)
  – Generalized SVD for Co-clustering (IEEE TKDE)
  – Spectral Clustering for Multi-type Relational Data (ICML 2006)
Consistent Bipartite Graph Copartitioning

- User graphs to represent the heterogeneous relationship.
- Divide the heterogeneous graph into a collection of bipartite graphs.
- Conduct spectral co-clustering on each bipartite graph, provided that the partitioning of the shared part of two bipartite graphs should be the same or almost the same.
- Develop an SDP-based solution to get the consistent partitioning results.
Consistent Partitioning

\[ A_{mon} \]

\[ M_1 = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \]

\[ D^{(1)} = \text{diag}(M_1 e) \]

\[ L^{(1)} = D^{(1)} - M_1 \]

\[ q = \begin{pmatrix} x \\ y \end{pmatrix}_{(m+n)d} \]

\[ B_{mot} \]

\[ M_2 = \begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix} \]

\[ D^{(2)} = \text{diag}(M_2 e) \]

\[ L^{(2)} = D^{(2)} - M_2 \]

\[ p = \begin{pmatrix} y \\ z \end{pmatrix}_{(n+r)d} \]

divide this tripartite graph into two bipartite graphs

partition these two graphs simultaneously and consistently
Formulating the Optimization Problem

• Minimize the cuts of the two bipartite graphs, with the constraints that their partitioning results on the central type of objects are the same.

• Objective Function:

\[
\begin{align*}
\min & \quad \frac{q^T L^{(1)} q}{q^T D^{(1)} q} \\
\min & \quad \frac{p^T L^{(2)} p}{p^T D^{(2)} p} \\
\text{subject to} & \quad q^T D^{(1)} e = 0, \quad q \neq 0 \\
& \quad p^T D^{(2)} e = 0, \quad p \neq 0 \\
& \quad 0 < \beta < 1
\end{align*}
\]

\[
q = \begin{pmatrix} x \\ y \end{pmatrix}_{(m+n) \times 1}
\]

\[
p = \begin{pmatrix} y \\ z \end{pmatrix}_{(n+t) \times 1}
\]
How to Solve the Optimization Problem #1: Convert it to a QCQP Problem

**Simplify the original Problem to single-objective programming**

\[
\begin{align*}
\min \beta \frac{q^T L^{(1)} q}{q^T D^{(1)} q} + (1 - \beta) \frac{p^T L^{(2)} p}{p^T D^{(2)} p} \\
\text{subject to} \quad q^T D^{(1)} e = 0, q \neq 0 \\
p^T D^{(2)} e = 0, p \neq 0 \\
0 < \beta < 1
\end{align*}
\]

**Sum-of-ratios Quadratic Fractional Programming**

\[
\begin{align*}
\min \left( \frac{\omega^T \Gamma_1 \omega}{\omega^T \Pi_1 \omega} + (1 - \beta) \frac{\omega^T \Gamma_2 \omega}{\omega^T \Pi_2 \omega} \right) \\
\text{subject to} \quad \omega^T \Pi_1 e = 0 \\
\omega^T \Pi_2 e = 0 \\
\omega \neq 0, 0 < \beta < 1
\end{align*}
\]

**Assistant Notations**

\[
\begin{align*}
\Gamma_1 &= \begin{bmatrix} L^{(1)} & 0 \\ 0 & 0 \end{bmatrix}_{s \times s}, \quad \Gamma_2 = \begin{bmatrix} 0 & 0 \\ 0 & L^{(2)} \end{bmatrix}_{s \times s} \\
\Pi_1 &= \begin{bmatrix} D^{(1)} & 0 \\ 0 & 0 \end{bmatrix}_{s \times s}, \quad \Pi_2 = \begin{bmatrix} 0 & 0 \\ 0 & D^{(2)} \end{bmatrix}_{s \times s}
\end{align*}
\]

**Quadratically Constrained Quadratic Programming (QCQP)**

\[
\begin{align*}
\min \omega^T \Gamma \omega \\
\text{subject to} \quad \omega^T \Pi_1 \omega = e^T \Pi_1 e \\
\omega^T \Pi_2 \omega = e^T \Pi_2 e \\
\omega^T \Pi_1 e = 0 \\
\omega^T \Pi_2 e = 0 \\
\Gamma = \frac{\beta}{e^T \Pi_1 e} \Gamma_1 + \frac{1 - \beta}{e^T \Pi_2 e} \Gamma_2, \quad 0 < \beta < 1
\end{align*}
\]
How to Solve the Optimization Problem #2: Convert QCQP to SDP

Semi-definite Programming (SDP)

\[
\begin{align*}
\min_{\omega, \Omega} & \quad \begin{bmatrix} 0 & 0 \\ 0 & \Gamma \end{bmatrix} \cdot \begin{bmatrix} 1 & \omega^T \\ \omega & \Omega \end{bmatrix} \\
\text{subject to} & \quad \begin{bmatrix} -e^T \Pi_1 e & 0 \\ 0 & \Pi_1 \end{bmatrix} \cdot \begin{bmatrix} 1 & \omega^T \\ \omega & \Omega \end{bmatrix} = 0 \\
& \quad \begin{bmatrix} -e^T \Pi_2 e & 0 \\ 0 & \Pi_2 \end{bmatrix} \cdot \begin{bmatrix} 1 & \omega^T \\ \omega & \Omega \end{bmatrix} = 0 \\
& \quad \begin{bmatrix} 0 & e^T \Pi_1/2 \\ \Pi_1 e/2 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & \omega^T \\ \omega & \Omega \end{bmatrix} = 0 \\
& \quad \begin{bmatrix} 0 & e^T \Pi_2/2 \\ \Pi_2 e/2 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & \omega^T \\ \omega & \Omega \end{bmatrix} = 0
\end{align*}
\]

\[
\begin{align*}
\min_w & \quad \begin{bmatrix} 0 & 0 \\ 0 & \Gamma \end{bmatrix} \cdot W \\
\text{subject to} & \quad \begin{bmatrix} -e^T \Pi_1 e & 0 \\ 0 & \Pi_1 \end{bmatrix} \cdot W = 0 \\
& \quad \begin{bmatrix} -e^T \Pi_2 e & 0 \\ 0 & \Pi_2 \end{bmatrix} \cdot W = 0 \\
& \quad \begin{bmatrix} 0 & e^T \Pi_1/2 \\ \Pi_1 e/2 & 0 \end{bmatrix} \cdot W = 0 \\
& \quad \begin{bmatrix} 0 & e^T \Pi_2/2 \\ \Pi_2 e/2 & 0 \end{bmatrix} \cdot W = 0 \\
& \quad \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \cdot W = 1, \\
& \quad \begin{bmatrix} 0 & e \\ e & 0 \end{bmatrix} \cdot W = \theta_1, \\
& \quad \begin{bmatrix} 0 & 0 \\ 0 & E \end{bmatrix} \cdot W = \theta_2 \\
W & \succeq 0
\end{align*}
\]
Extension to More Complex Heterogeneous Graphs

\[
\begin{align*}
\min \sum_{i=1}^{k-1} \beta_i \frac{q_i^T L^{(i)} q_i}{q_i^T D^{(i)} q_i} \\
\text{subject to} \quad q_i^T D^{(i)} e = 0, \quad q_i \neq 0, \quad i = 1, \ldots, k-1 \\
\sum_{i=1}^{k-1} \beta_i = 1, \quad 0 < \beta_i < 1
\end{align*}
\]
Consistent Information-theoretic Co-clustering

The consistency concept

Information-theoretic coclustering (KDD 2003)
Mathematical Formulation

• Co-clustering
  \[C_X : \{x_1, ..., x_m\} \rightarrow \{\hat{x}_1, ..., \hat{x}_r\}\]
  \[C_Y : \{y_1, ..., y_n\} \rightarrow \{\hat{y}_1, ..., \hat{y}_s\}\]
  \[C_Z : \{z_1, ..., z_l\} \rightarrow \{\hat{z}_1, ..., \hat{z}_t\}\]

• A consistent co-clustering minimizes the following objective functions

  \[(i) \quad F(X,Y,Z) = \alpha D(p_1(X,Y) \parallel q_1(X,Y)) + (1 - \alpha)D(p_2(Y,Z) \parallel q_2(Y,Z)),\]
  \[\text{where } 0 < \alpha < 1\]

  \[(ii) \quad F(X,Y,Z) = \min_{X,Y,Z} \left\{ \max \left\{ D(p_1(X,Y) \parallel q_1(X,Y)), D(p_2(Y,Z) \parallel q_2(Y,Z)) \right\} \right\}\]

• Similar iterative method can be used to optimize \(F(X,Y,Z)\), and the convergence can be proved.
Generalized SVD for Co-clustering

• Rather than integrating heterogeneous relationship in a unified matrix or using tensor, we try to connect heterogeneous relationships using generalized SVD.

• While SVD corresponds to the optimal embedding of bipartite graph, GSVD might correspond to tripartite graph.

\[ \begin{align*}
\text{Theorem 1} & \text{ If we have } \hat{A} \in \mathbb{R}^{m \times n} \text{ and } \hat{B} \in \mathbb{R}^{n \times d}, m \leq n \leq t, \text{ then there exists unitary matrices } U \in \mathbb{R}^{m \times m}, V \in \mathbb{R}^{n \times n}, \text{ and reversible matrix } X \in \mathbb{R}^{n \times n} \text{ such that:} \\
& \quad \begin{bmatrix} \hat{A} = U C X^T \\ \hat{B} = X S V^T \end{bmatrix}, \\
& \text{where } C = \text{diag}(c_1, c_2, \ldots, c_m), c_i \geq 0 \text{ and } S = \text{diag}(s_1, s_2, \ldots, s_n), s_i \geq 0. \end{align*} \]
Generalized SVD for Co-clustering

1. Given $A$ and $B$, form $P_1$, $P_2$, $R_1$, $R_2$, and $\hat{A}$, $\hat{B}$.
2. Compute GSVD of $\hat{A}$, $\hat{B}$ to get $U$, $X$, $V$, $C$, and $S$.
3. Form $H = CX^TXS$ and compute SVD of it to get $U_H$, $V_H$.
4. Form $U^* = UU_H$, $V^* = VV_H$ and take the second column vectors of them, $u_2$ and $v_2$, to form the normalized embedding vector

$$\omega_2 = [P_1^{-1/2}u_2 \quad R_2^{-1/2}v_2]^T.$$ 

5. Cluster on the one-dimensional data $P_1^{-1/2}u_2$ and $R_2^{-1/2}v_2$ to obtain the desired bipartition of categories and terms, respectively.

No mathematical proof yet, since generalized SVD has no explicit objective function.
Spectral Clustering for Multi-type Relational Data

- Handling both pairwise relations and features

\[
L = \sum_{1 \leq i < j \leq m} w_a^{(ij)} \| R^{(ij)} - C^{(i)} A^{(ij)} (C^{(j)})^T \|^2 + \sum_{1 \leq i \leq m} w_b^{(i)} \| F^{(i)} - C^{(i)} B^{(i)} \|^2
\]

\[
\max_{\{(C^{(i)})^T C^{(i)} = I_{k_i}\} 1 \leq i \leq m} \sum_{1 \leq i < j \leq m} w_b^{(ij)} \text{tr}((C^{(i)})^T F^{(i)} (F^{(i)})^T C^{(i)}) + \sum_{1 \leq i < j \leq m} w_a^{(ij)} \text{tr}((C^{(i)})^T R^{(ij)} C^{(j)} (C^{(j)})^T (R^{(ij)})^T C^{(i)})
\]

\[
\max_{(C^{(p)})^T C^{(p)} = I_{k_p}} \text{tr}((C^{(p)})^T M^{(p)} C^{(p)})
\]

\[
M^{(p)} = w_b^{(p)} (F^{(p)} (F^{(p)})^T) + \sum_{p < j \leq m} w_a^{(pj)} (R^{(pj)} C^{(j)} (C^{(j)})^T (R^{(pj)})^T) + \sum_{1 \leq j < p} w_a^{(jp)} ((R^{(jp)})^T C^{(j)} (C^{(j)})^T (R^{(jp)})^T).
\]
Optimization Steps

- It can be proved the final equivalent optimization problem has close-form solution.
- The following algorithm is used to approximate this solution.

**Algorithm 1** Spectral Relational Clustering

| Input: Relation matrices $\{R^{(ij)} \in \mathbb{R}^{n_i \times n_j}\}_{1 \leq i < j \leq m}$, feature matrices $\{F^{(i)} \in \mathbb{R}^{n_i \times f_i}\}_{1 \leq i \leq m}$, numbers of clusters $\{k_i\}_{1 \leq i \leq m}$, weights $\{w_a^{(ij)}, w_b^{(ij)} \in R^{-}\}_{1 \leq i < j \leq m}$. |
| Output: Cluster indicator matrices $\{C^{(p)}\}_{1 \leq p \leq m}$. |

**Method:**

1. Initialize $\{C^{(p)}\}_{1 \leq p \leq m}$ with orthonormal matrices.
2. repeat
3. for $p = 1$ to $m$ do
4. Compute the matrix $M^{(p)}$ as in Eq. (9).
5. Update $C^{(p)}$ by the leading $k_p$ eigenvectors of $M^{(p)}$.
6. end for
7. until convergence
8. for $p = 1$ to $m$ do
9. transform $C^{(p)}$ into a cluster indicator matrix by the k-means.
10. end for
Discussions on Collective Graphs

• Pros
  – It is more natural to decompose heterogeneous relationships into homogenous relationships, than to combine homogeneous relationships to heterogeneous relationships.

• Cons
  – Complexity of graph processing is relatively high than power method.
  – Graph fusion has not been well studied yet.
## Summary

<table>
<thead>
<tr>
<th></th>
<th>URM</th>
<th>Tensor</th>
<th>Consistent Bipartite Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clustering</td>
<td>SimFusion</td>
<td>CubeSVD</td>
<td>Consistent Bipartite Graph Copartitioning</td>
</tr>
<tr>
<td>Clustering</td>
<td>Multi-type LSA</td>
<td></td>
<td>Consistent Information-Theoretic Coclustering for Multi-Data</td>
</tr>
<tr>
<td>Ranking</td>
<td>LinkFusion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ranking</td>
<td>Object-level Ranking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ranking</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CubeRank?  
Consistent Rank?
Future Work

• Modeling the heterogeneous relationship more effectively.
  – Matrix, tensor, graphs, ...
  – What is the next?

• Develop more efficient algorithms for high-order heterogeneous data mining.
  – Scalability is an issue for most of the algorithms mentioned in this talk.
  – Large-scale (multi-)linear algebra and large scale optimization
  – Supervised or semi-supervised learning for high-order heterogeneous data (i.i.d is not a reasonable assumption).
Further Discussions

• Although data objects are heterogeneous, they can be regarded as sampled from the same probability space.
  – The heterogeneity just comes from different views of the space.
• Can we recover the unified probability space and solve this problem from the root?
  – Reference paper
Thanks!

tyliu@microsoft.com
http://research.microsoft.com/users/tyliu/