Clustered Low Rank Approximation and Applications to Social Network Analysis

Donghyuk Shin
University of Texas at Austin
Department of Computer Science

Nov 3, 2011

Joint work with Inderjit Dhillon and Si Si

Outline

- **Social Network Analysis**
  - Two Applications
    - Finding Influential Users
    - Link Prediction
  - Matrix Functions
  - Low rank approximation
- **Proposed Methods**
  - Clustered low rank approximation
  - Hierarchical approach
- **Experimental Results**
- **Summary and Conclusions**

Social Network Analysis

Networks

- Social networks (Facebook, MySpace, LiveJournal, ...)
- Communication networks (SKT)
- Biological networks (linking genes to diseases)

Applications

- Link prediction — who will become whose friend in the future
- Finding influential people (centrality)
- Proximity measures — "distances" between pairs of vertices
- Community detection

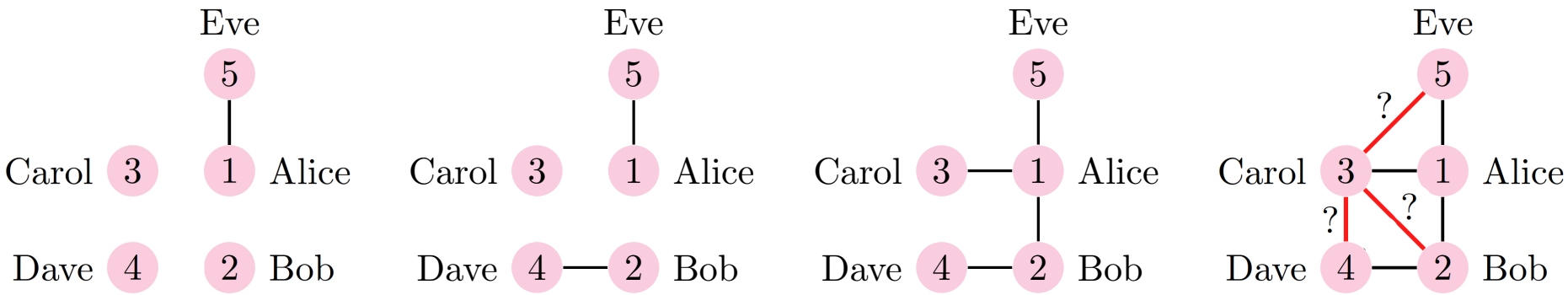
Challenges

- Massive scale — Facebook's social graph has 700 million vertices
- Multiple sources of information
- Need scalable algorithms and "good" predictions

Link Prediction Problem

Consider a social network that evolves with time,

$$\ldots \longrightarrow A^{(t-2)} \longrightarrow A^{(t-1)} \longrightarrow A^{(t)} \xrightarrow{?} A^{t+1}$$



Q: Can we predict links that will form at time step $t + 1$?

A: Many models exist, e.g. common neighbors, the Katz measure, etc.

Finding Influential People

Problem setting:

- Given
  - a limited budget B for initial advertising (eg. free samples of product)
  - estimates for influence between individuals
- Goal
  - trigger a large cascade of influence (eg. further adoptions of a product)
- Question
  - which set of individuals should B target at?

What we need:

- models of influence in social networks.
- algorithms to maximize spread of influence.

Models of Influence

- Independent Cascade
  - node v becomes active $\rightarrow$ single chance of activating inactive neighbor u.
  - activation attempt succeeds with probability p (diffusion speed).
  - diffusion process stops when there are no newly activated nodes.
- Linear Threshold
  - node v has random threshold $\theta_v \sim U[0,1]$
  - node v is influenced by each neighbor u according to a weight $b_{v,u}$

$$\sum_{u \text{ neighbor of } v} b_{v,u} \leq 1.$$

  - node v becomes active when

$$\sum_{u \text{ active neighbor of } v} b_{v,u} \geq \theta_v.$$

Influence Maximization Problem

Influence spread of node set S: $g(S)$

- expected number of active nodes at the end, if S is the initial active set.

Properties of $g(S)$

- non-negative & monotone: $g(S \cup v) \geq g(S)$
- submodular: $\forall S \subset T$, $\forall v \in V \setminus T$

$$g(S \cup v) - g(S) \geq g(T \cup v) - g(T)$$

Influence Maximization

- Given a parameter k (budget), find a k-node set S to maximize $g(S)$.
- Constrained discrete optimization problem (NP-hard).

D. Kempe, J. Kleinberg and E. Tardos. *Maximizing the spread of influence through a social network*. SIGKDD, 2003.

Greedy Algorithm

Greedy algorithm gives an $(1 - 1/e)$-approximation:

- start with an empty set S
- for k iterations:
  - add node v to S that maximizes $g(S \cup v) - g(S)$

Can't scale up to large problems.
$\rightarrow$ use matrix functions $f(A)$ to approximate $g(S)$.

Proposed method:

- initialize $S = \emptyset$
- for $r = 1$ to k do
  - select $v = \arg\max_{i \in V \setminus S}(\sum_j f(A)_{ij} w_j)$
  - $S = S \cup \{v\}$
  - remove node v from A

Matrix Functions

- The adjacency matrix $A = [a_{ij}]$ of graph $G = (\mathcal{V}, \mathcal{E})$ is given by

$$a_{ij} = \begin{cases} w_{ij}, & \text{if there is an edge between nodes } i \text{ and } j \\ 0, & \text{otherwise} \end{cases}$$

- Powers of A:
  $(A^n)_{ij}$ counts the number of length-n walks between nodes i and j
- Matrix Functions serve as useful conceptual and computational tool, e.g. matrix exponential and Katz measure:

$$\begin{aligned} f(A) &= e^A = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots + \frac{A^k}{k!} + \cdots \\ f(A) &= (I - \beta A)^{-1} = I + \beta A + \beta^2 A^2 + \beta^3 A^3 + \cdots \end{aligned}$$

E. Estrada and D. J. Higham. *Network Properties Revealed through Matrix Functions*. SIAM Review, 2010.

Matrix Functions

- Centrality: $f(A)_{ii}$, quantifies node i's "well-connectedness"
- Communicability: $f(A)_{ij}$, measures how easy it is for information to pass from node i to j
- Betweenness: quantifies influence of a node as information flows around a network, how communicability changes when a node is removed

$$\sum \sum_{i \neq j, i \neq r, j \neq r} \frac{f(A)_{ij} - f(A - E(r))_{ij}}{f(A)_{ij}}$$

Low-Rank Approximations

Truncated SVD:

$$A^i \approx V \Lambda^i V^{\mathsf{T}}$$

allows approximations of matrix functions:

$$\begin{aligned} f(A) &\approx V f(D) V^T \\ e^A &\approx V e^{\Lambda} V^T \\ (I - \beta A)^{-1} &\approx V (I - \beta \Lambda)^{-1} V^T \end{aligned}$$

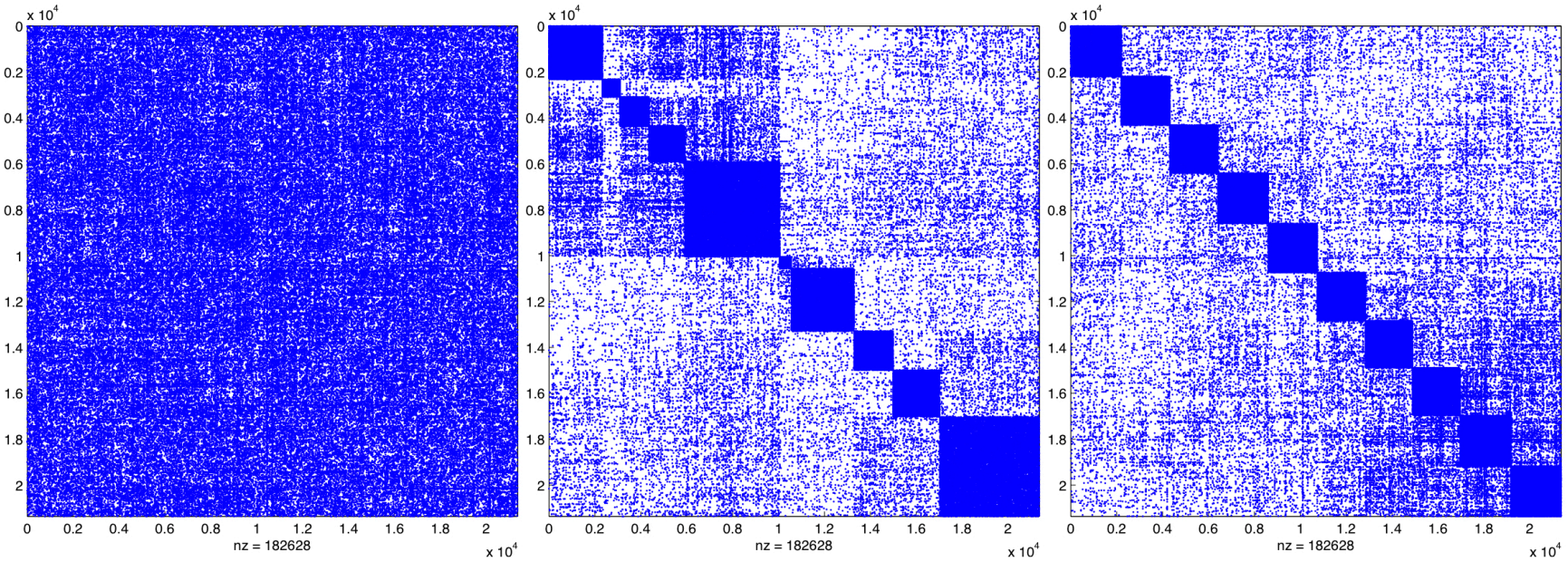
BUT SVD has Major Drawback. For example, in LiveJournal graph:

- 3.8 million nodes and 65 million edges (avg. degree $= 4.5$)
- Requires about 1GB of memory to store adjacency matrix
- Rank-100 approximation requires about 5.7GB of memory

PROBLEM: SVD Does Not Preserve SPARSITY

Proposed Method

Clustering example: arXiv data graph **condMat**



# of clust	edges in clust	# of vertices	# of edges	relative error
10 (graclus)	79.8 %	21,363	182,628	45.0
10 (metis)	77.9 %	21,363	182,628	47.0

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1c} \\ \vdots & \ddots & \vdots \\ A_{c1} & \cdots & A_{cc} \end{bmatrix}$$

Clustered low rank approximation

Clustered low rank approximation (CLRA):

- Cluster the graph into c clusters
- Compute a low rank approximation of each cluster:

$$A_{ii} = U_i S_i U_i^T$$

- Extend the cluster-wise approximations, into an approximation for the entire matrix

$$S_{ij} = U_i^T A_{ij} U_j$$

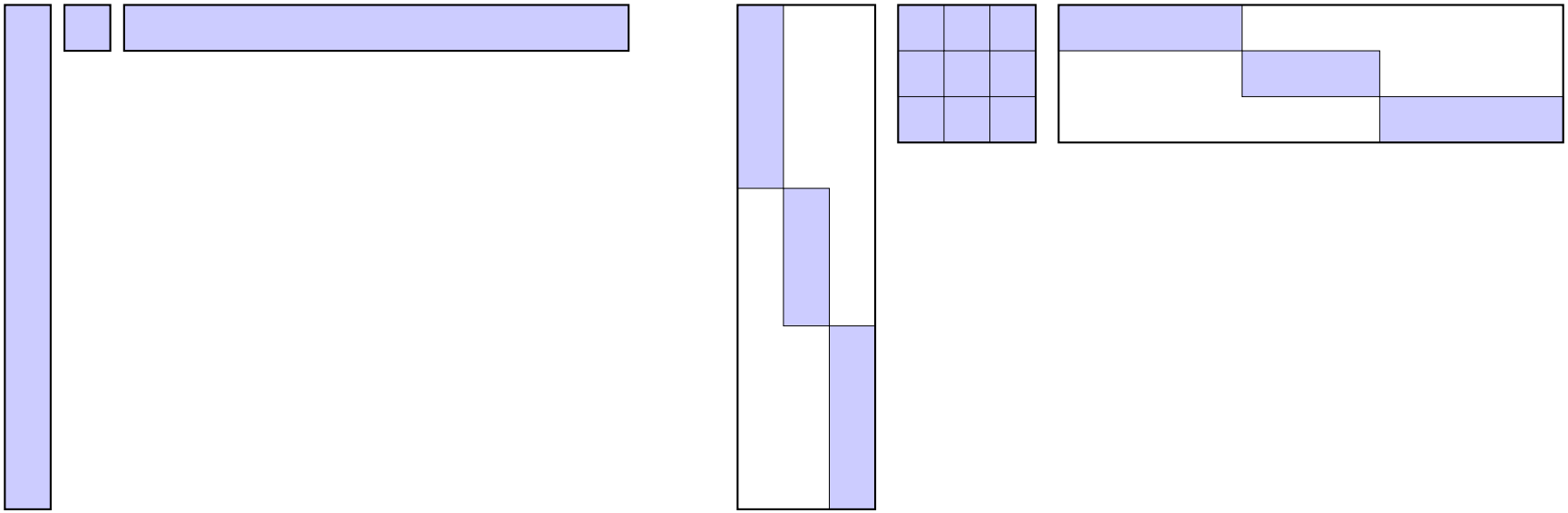
$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \approx \begin{bmatrix} U_{11} & 0 \\ 0 & U_{22} \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} U_{11}^T & 0 \\ 0 & U_{22}^T \end{bmatrix}$$

B. Savas and I. S. Dhillon. *Clustered low rank approximation of graphs in information science applications*. SDM, 2011.

Low rank vs clustered low-rank

Low rank: $A \approx U \Sigma V^{\mathsf{T}}$

Clustered low rank: $A \approx \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix}^{\mathsf{T}}$



- Observe $\mathrm{diag}(U_1, U_2, \cdots, U_c)$ and has the same memory usage as U
- Experiments show that most of U is contained in $\mathrm{diag}(U_1, U_2, \cdots, U_c)$
- Similarly of V and $\mathrm{diag}(V_1, V_2, \cdots, V_c)$

Approximations within each cluster

- Deterministic methods — truncated SVD using ARPACK, PROPACK, SVDPACK.
- Stochastic methods

$$Y = A\Omega \qquad \text{or} \qquad Y = (AA^T)^q A\Omega$$

where q is small (1,2 or 3), and Ω is a random matrix (standard Gaussian). An approximation is obtained with

$$A \approx \hat{A} = P_Y A = Y(Y^T Y)^{-1} Y^T A$$

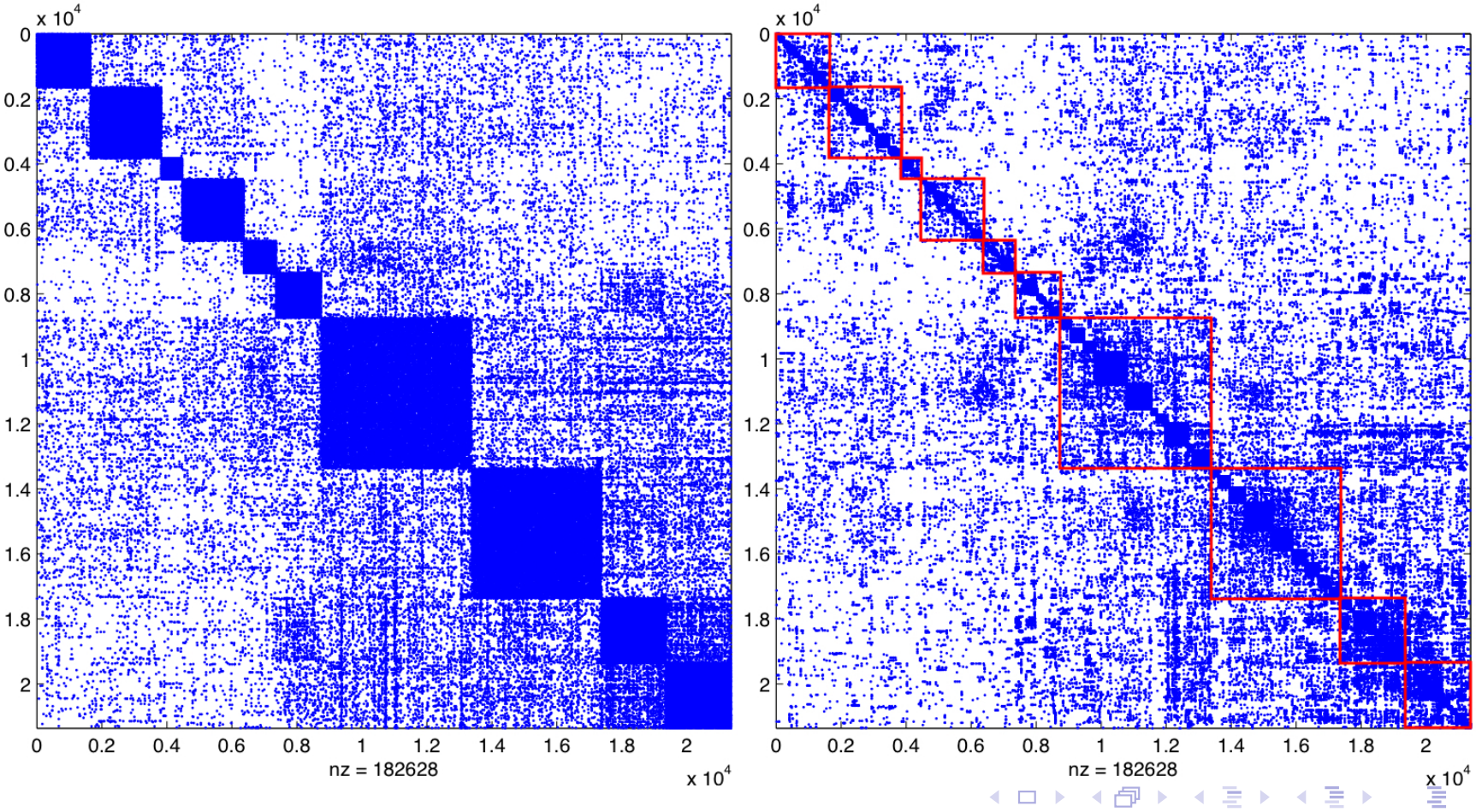
where P_Y is an orthogonal projection onto the range space of Y

N. Halko, P. G. Martinsson, and J. A. Tropp. *Finding Structure with Randomness*. SIAM Review, 2011.

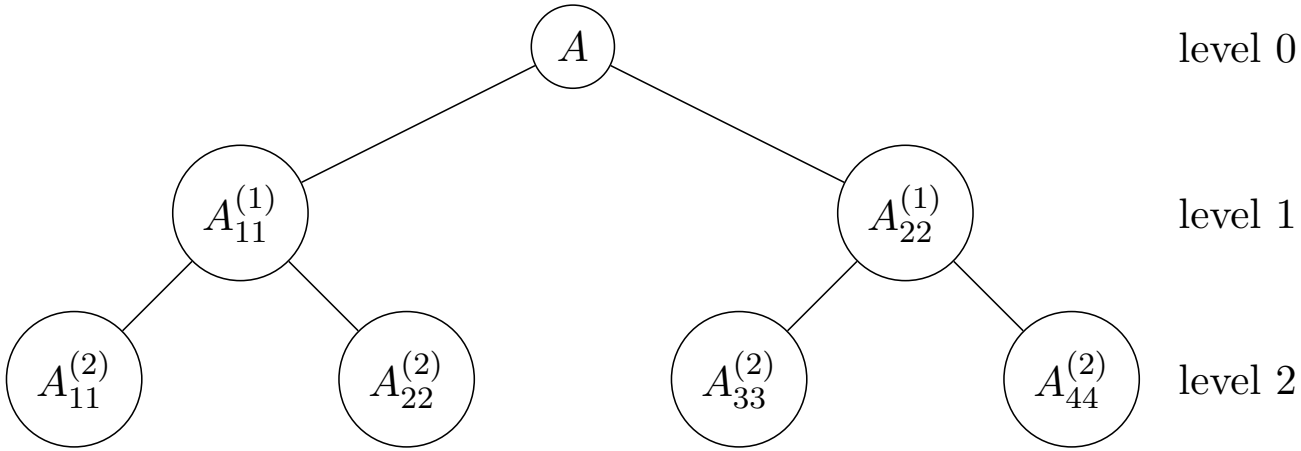
Hierarchical Clustering: arXiv data graph **condMat**

Networks exhibit a hierarchical structure.

- exploit hierarchical structure.
- predictions at multiple scales.



Hierarchical CLRA



$$A = \begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} \\ A_{21}^{(1)} & A_{22}^{(1)} \end{bmatrix}, \quad A_{11}^{(1)} = \begin{bmatrix} A_{11}^{(2)} & A_{12}^{(2)} \\ A_{21}^{(2)} & A_{22}^{(2)} \end{bmatrix}, \quad A_{22}^{(1)} = \begin{bmatrix} A_{33}^{(2)} & A_{34}^{(2)} \\ A_{43}^{(2)} & A_{44}^{(2)} \end{bmatrix}$$

Clustered low rank on bottom level:

$$A_{11}^{(1)} \approx \begin{bmatrix} U_1^{(2)} & 0 \\ 0 & U_2^{(2)} \end{bmatrix} \begin{bmatrix} S_{11}^{(2)} & S_{12}^{(2)} \\ S_{21}^{(2)} & S_{22}^{(2)} \end{bmatrix} \begin{bmatrix} U_1^{(2)} & 0 \\ 0 & U_2^{(2)} \end{bmatrix}^T$$

$$A_{22}^{(1)} \approx \begin{bmatrix} U_3^{(2)} & 0 \\ 0 & U_4^{(2)} \end{bmatrix} \begin{bmatrix} S_{33}^{(2)} & S_{34}^{(2)} \\ S_{43}^{(2)} & S_{44}^{(2)} \end{bmatrix} \begin{bmatrix} U_3^{(2)} & 0 \\ 0 & U_4^{(2)} \end{bmatrix}^T$$

Hierarchical CLRA

Q: How to get $U_1^{(1)}$?

- A1: do SVD or stochastic method on $A_{11}^{(1)}$
  $\Rightarrow$ slow — most time consuming part of CLRA

Subspace spanned by bases of each level should be close

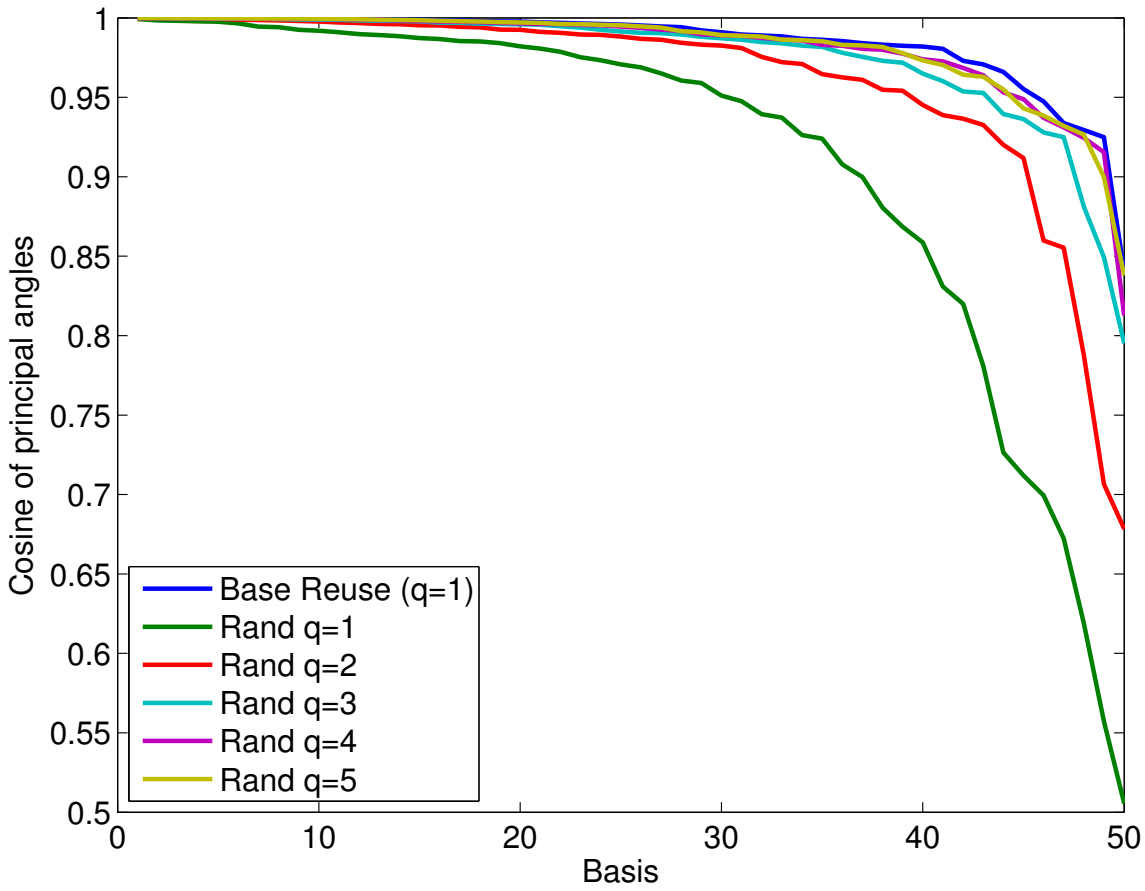
- A2: Stochastic method with $\Omega = \begin{bmatrix} U_1^{(2)} & 0 \\ 0 & U_2^{(2)} \end{bmatrix}$ instead of a random matrix.
  $\Rightarrow$ faster and more accurate approximation.

Complete approximation with

$$S_{ij}^{(1)} = {U_i^{(1)}}^T A_{ij} U_j^{(1)}$$

Closeness to SVD: Principal Angles

- arXiv data graph **AstroPh**: 17,903 nodes, 394,003 edges.



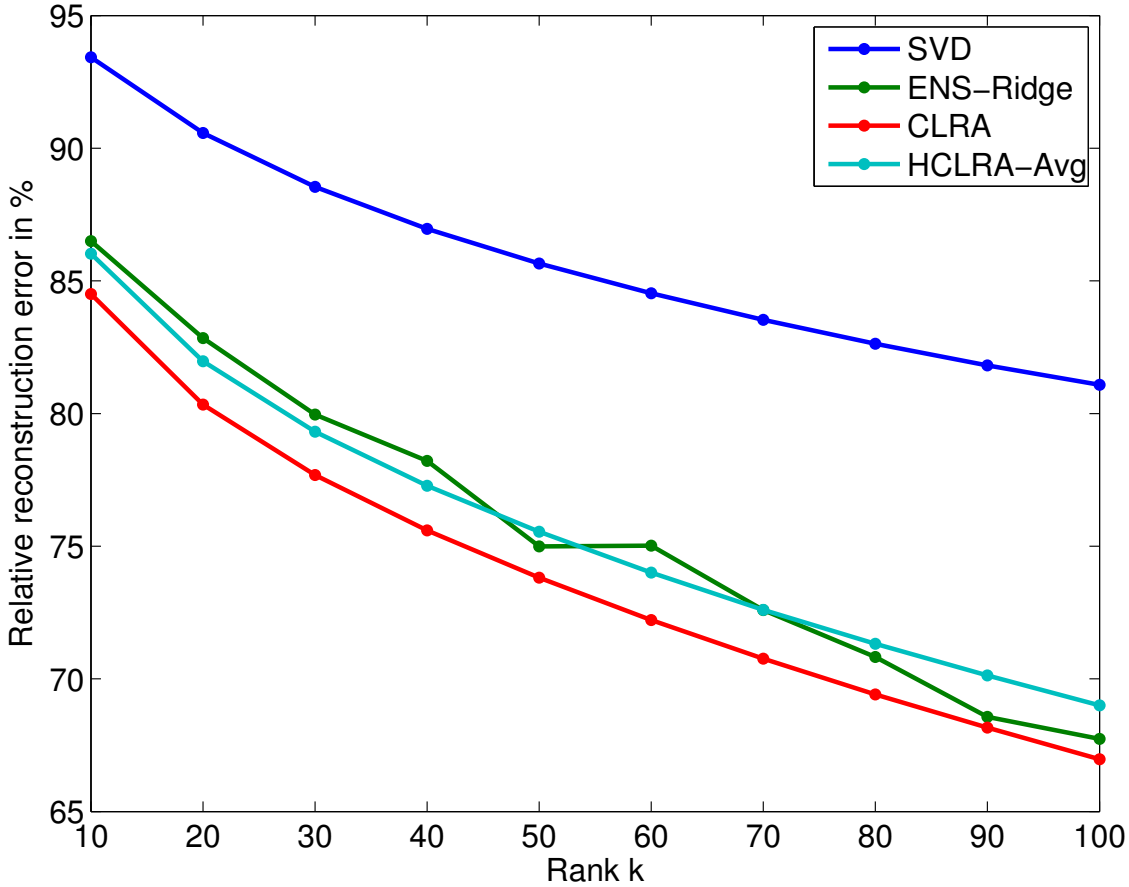
Prediction with Hierarchical CLRA

1. Do hierarchical clustering on the graph
2. Compute low rank approximation of bottom level and make predictions
3. For each upper level k:
   - compute low rank approximation using $(k+1)$-th level
   - make predictions
4. Combine predictions from all levels and make final predictions
   - Influential users: take average of $g(S)$ over all levels
   - Link prediction: take average of Katz score over all levels

Experimental Results

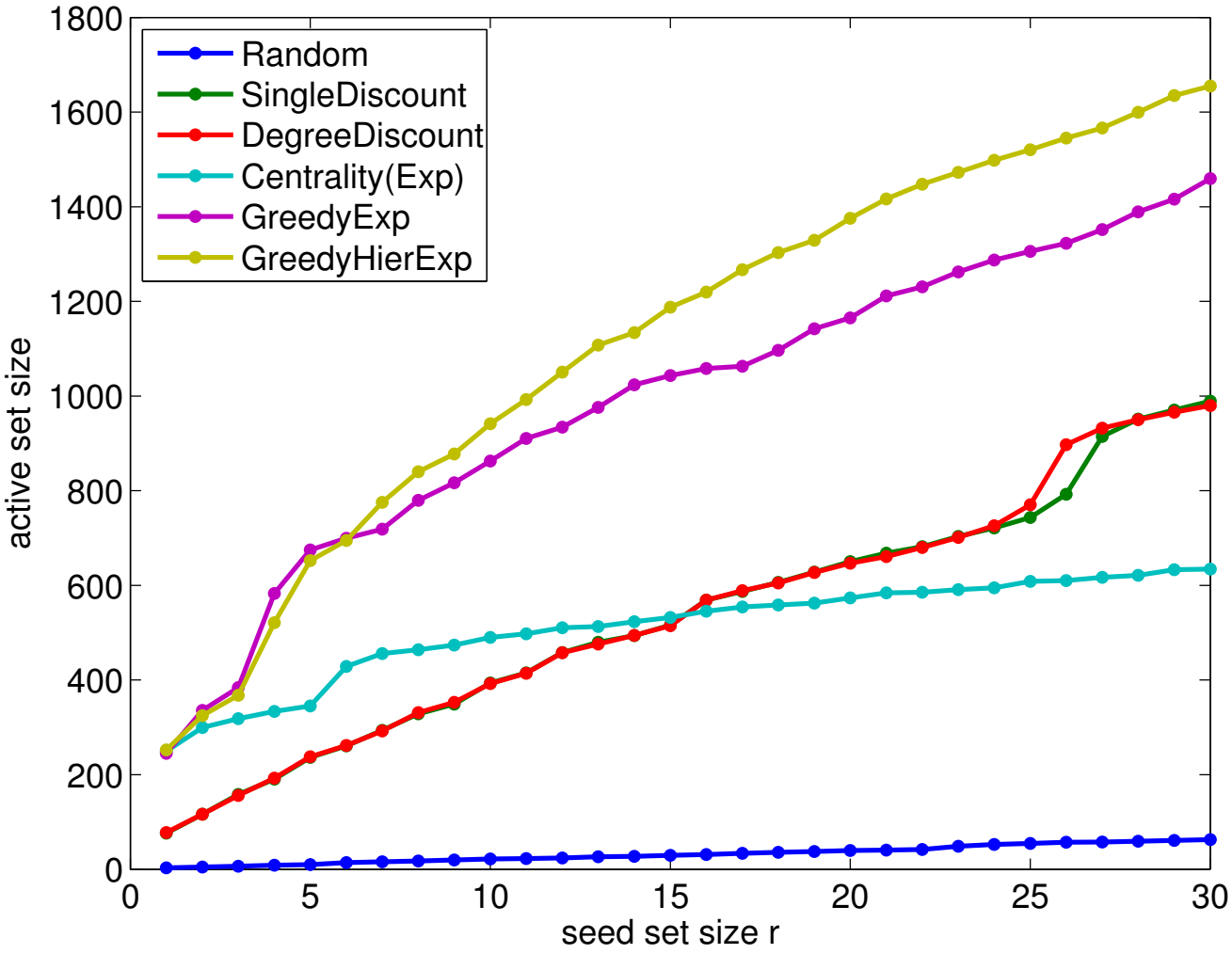
Relative Error

- LiveJournal subgraph: 13,556 nodes, 365,272 edges.
- 3 levels, 2 clusters per level



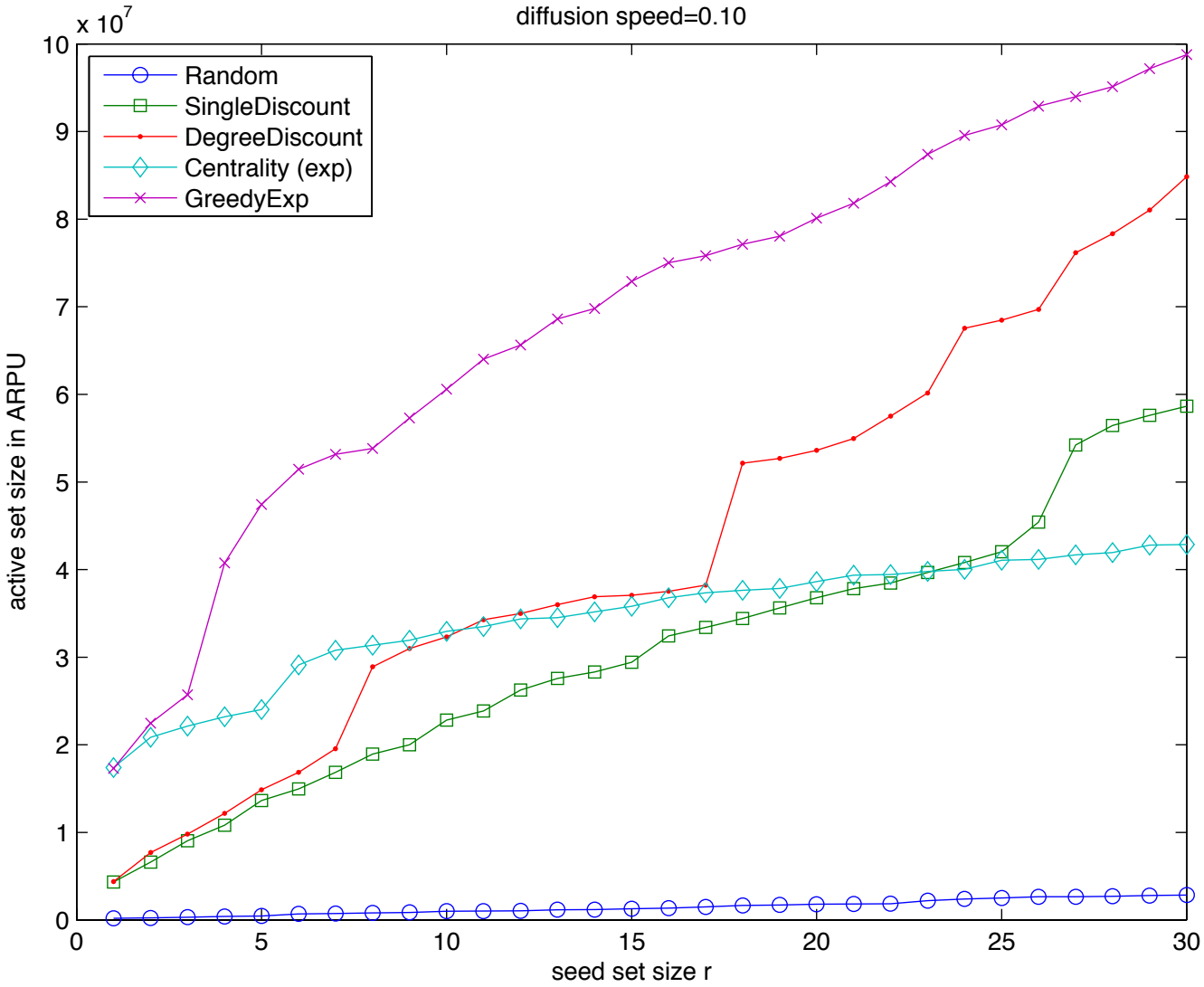
Influence Spread

SKT network (1.3M nodes 3.1M edges)

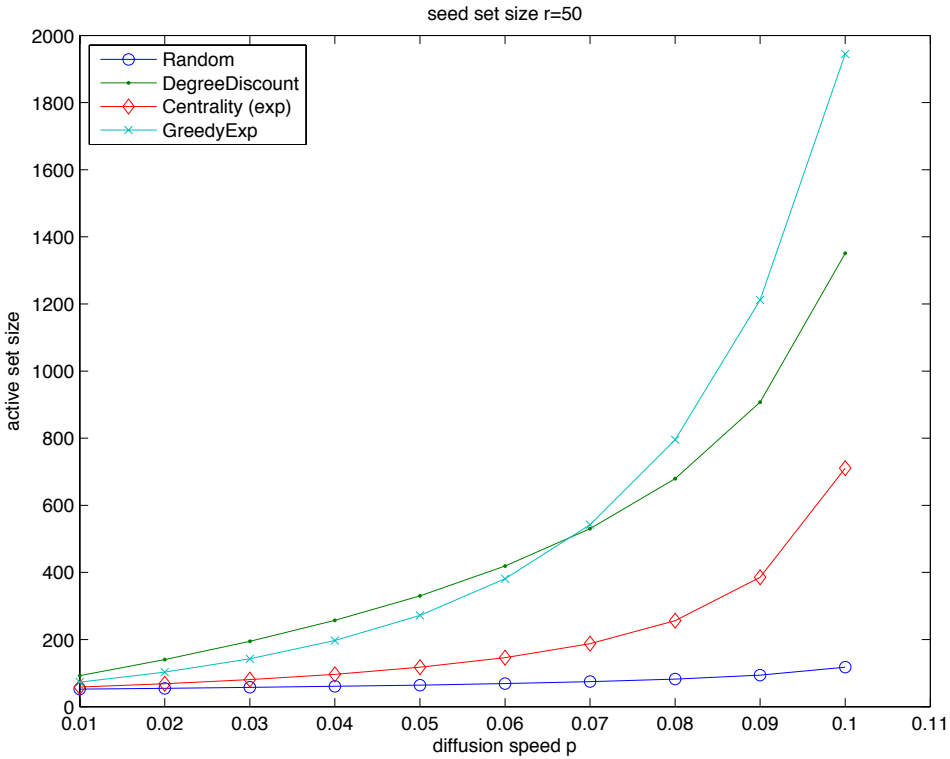


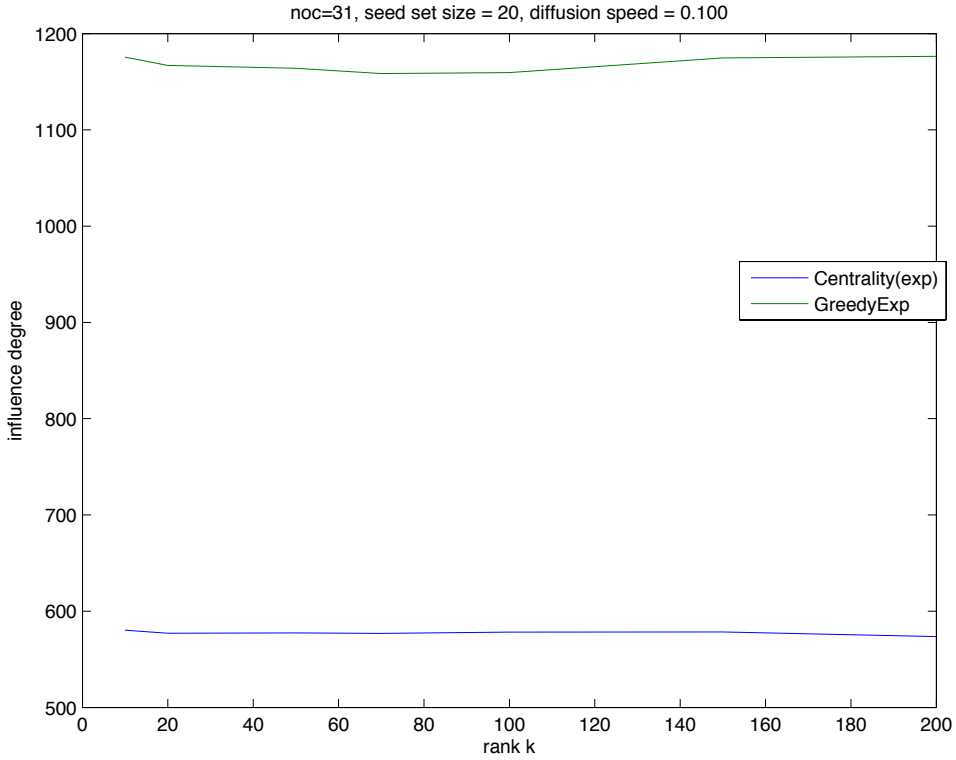
Weighted Influence Spread

SKT network (1.3M nodes 3.1M edges)



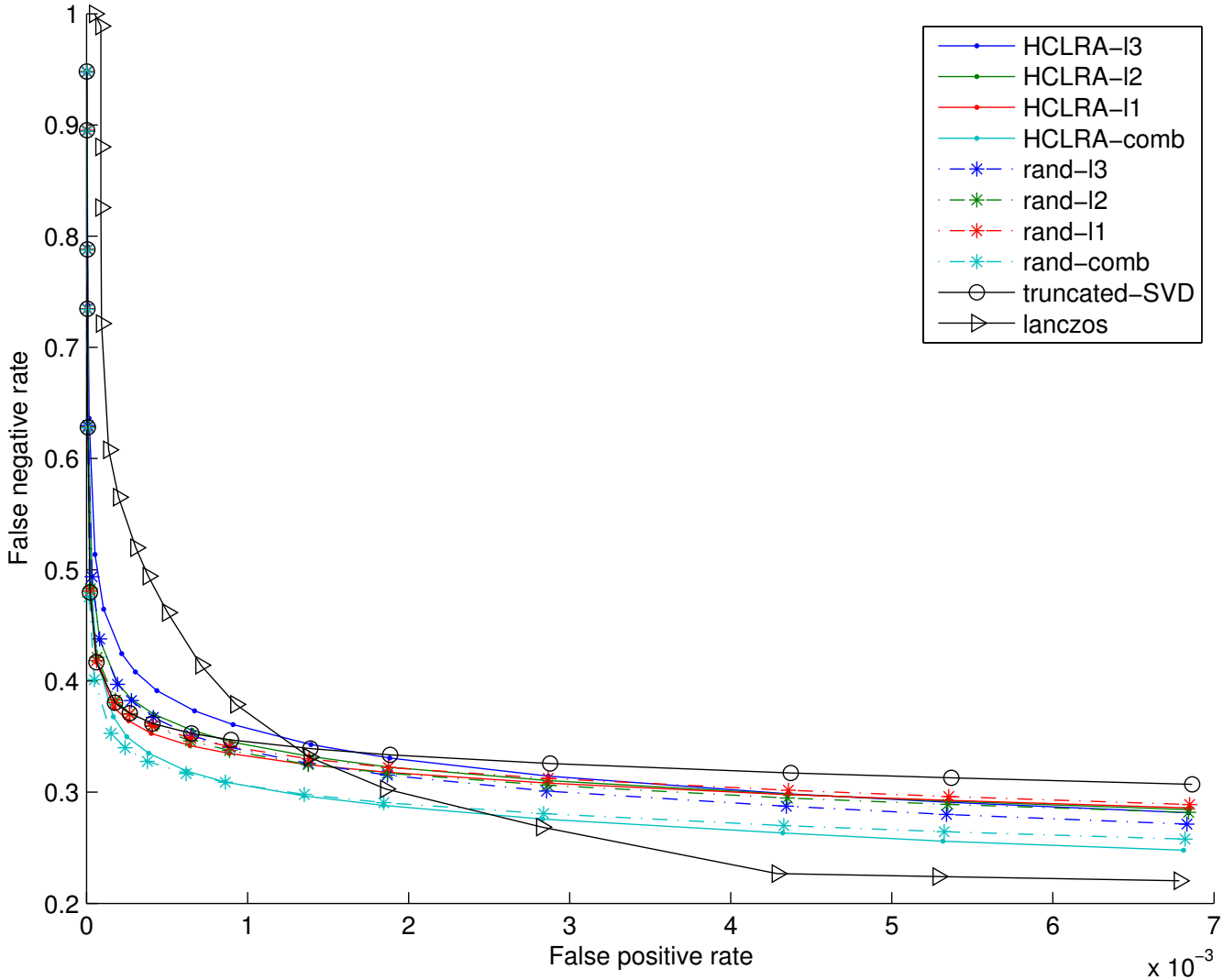
Varying p and k





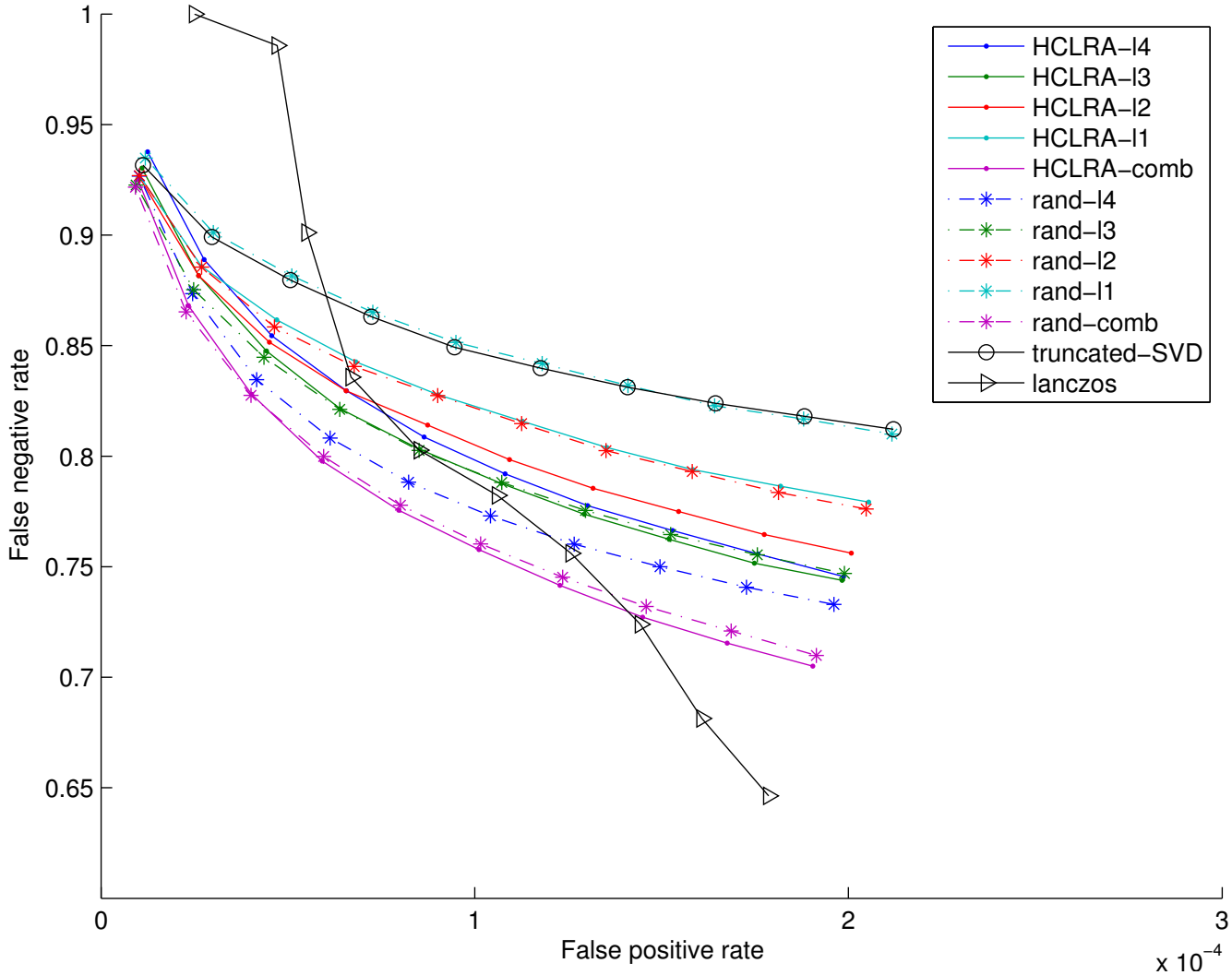
Link Inference

ca-HepPh (N=11,204)



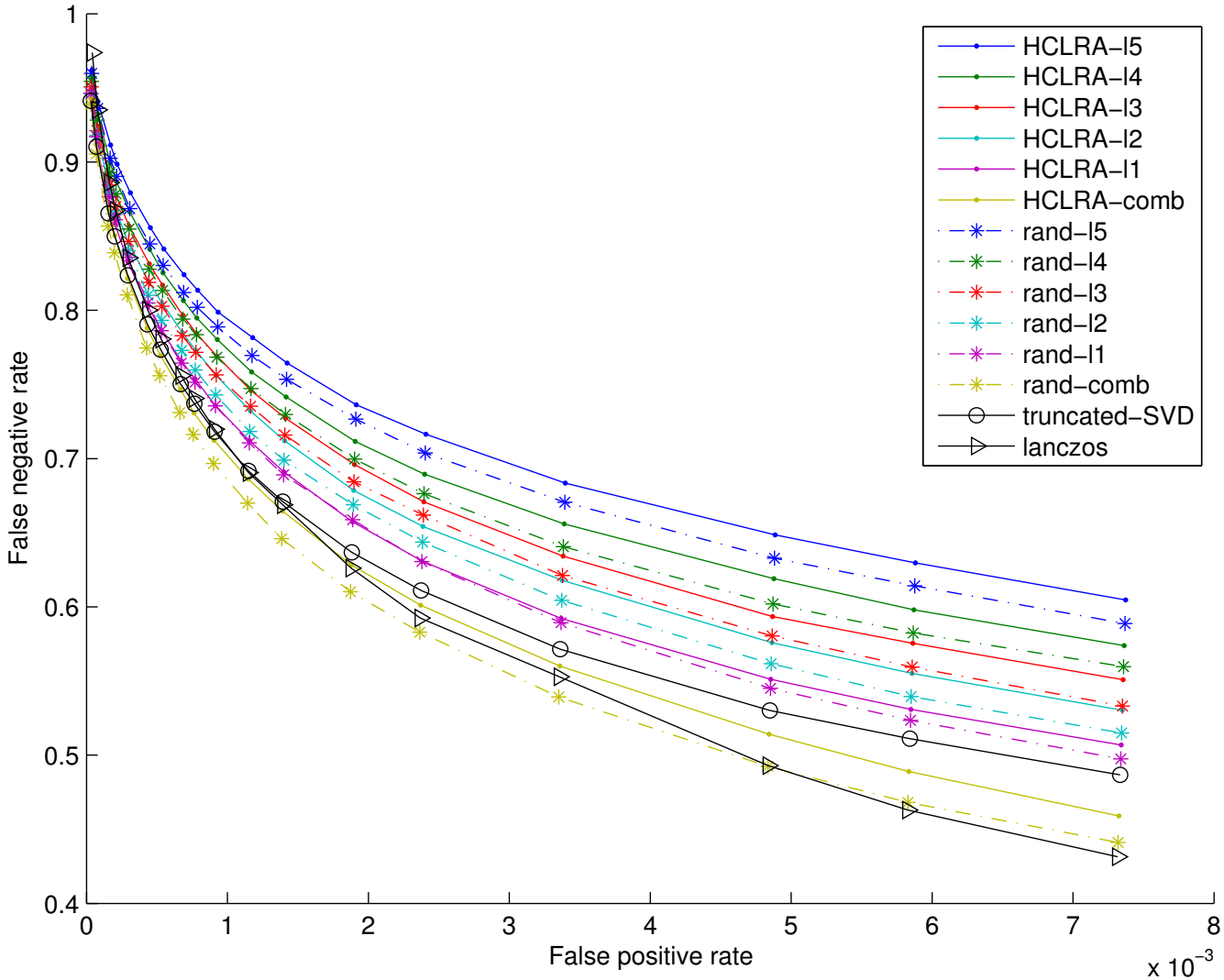
Link Inference

ca-CondMat (N=21,363)



Link Inference

Epinions (N=32,223)



Summary and Conclusions

- Structural Analysis of Massive Social Networks
- New Method — Hierarchical clustered low rank approximation
  - Improved quality of approximation
  - Improved performance by combining multilevel predictions
  - Construct bases from lower level — no additional memory
- Applied to computation of proximity measures & link prediction on large social networks
- Future Work
  - Supervised learning — approximation, link prediction
  - Better strategy of combining predictions

References

B. Savas and I. S. Dhillon. *Clustered low rank approximation of graphs in information science applications*. SIAM Conference on Data Mining, 2011.

I. S. Dhillon, Y. Guan and B. Kulis. *Weighted graph cuts without eigenvectors a multilevel approach*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007.

E. Estrada and D. J. Higham. *Network Properties Revealed through Matrix Functions*. SIAM Review, 2010.

D. Easley and J Kleinberg, *Networks, Crowds, and Markets*, Cambridge University Press, 2010.

D. Kempe, J. Kleinberg and E. Tardos, *Maximizing the spread of influence through a social network*, SIGKDD, 2003.