On social networks and collaborative recommendation

Social Networks WS09/10
Speaker: Julia Wolf

based on a paper by Ioannis Konstas, Vassilios Stathopoulos and Joemon M Jose

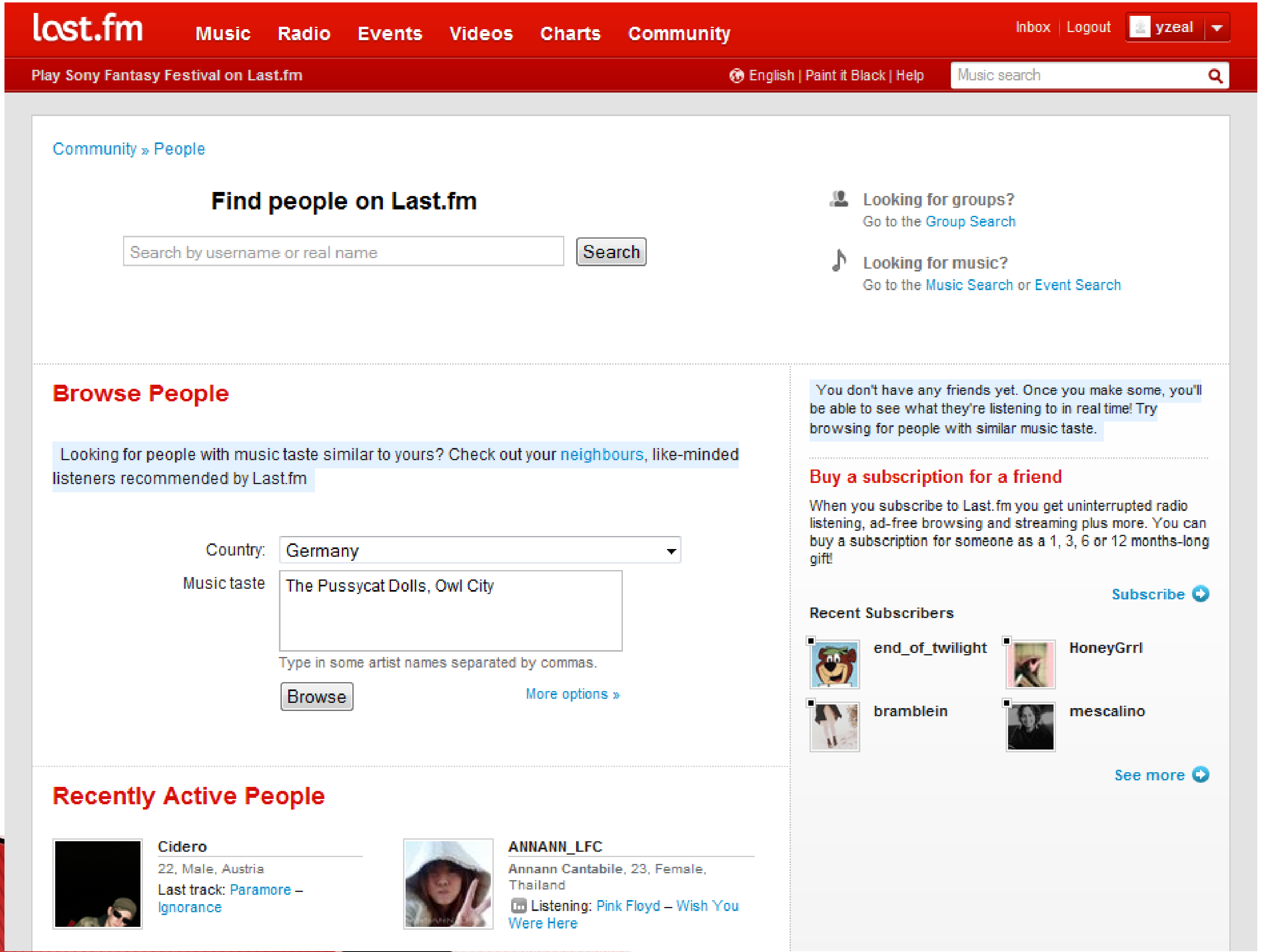
Outline

- The Network
- Collaborative Recommendation
- Collaborative Filtering
- The Social Graph
- Random Walks with Restarts
- Experiments and Results
- Conclusion

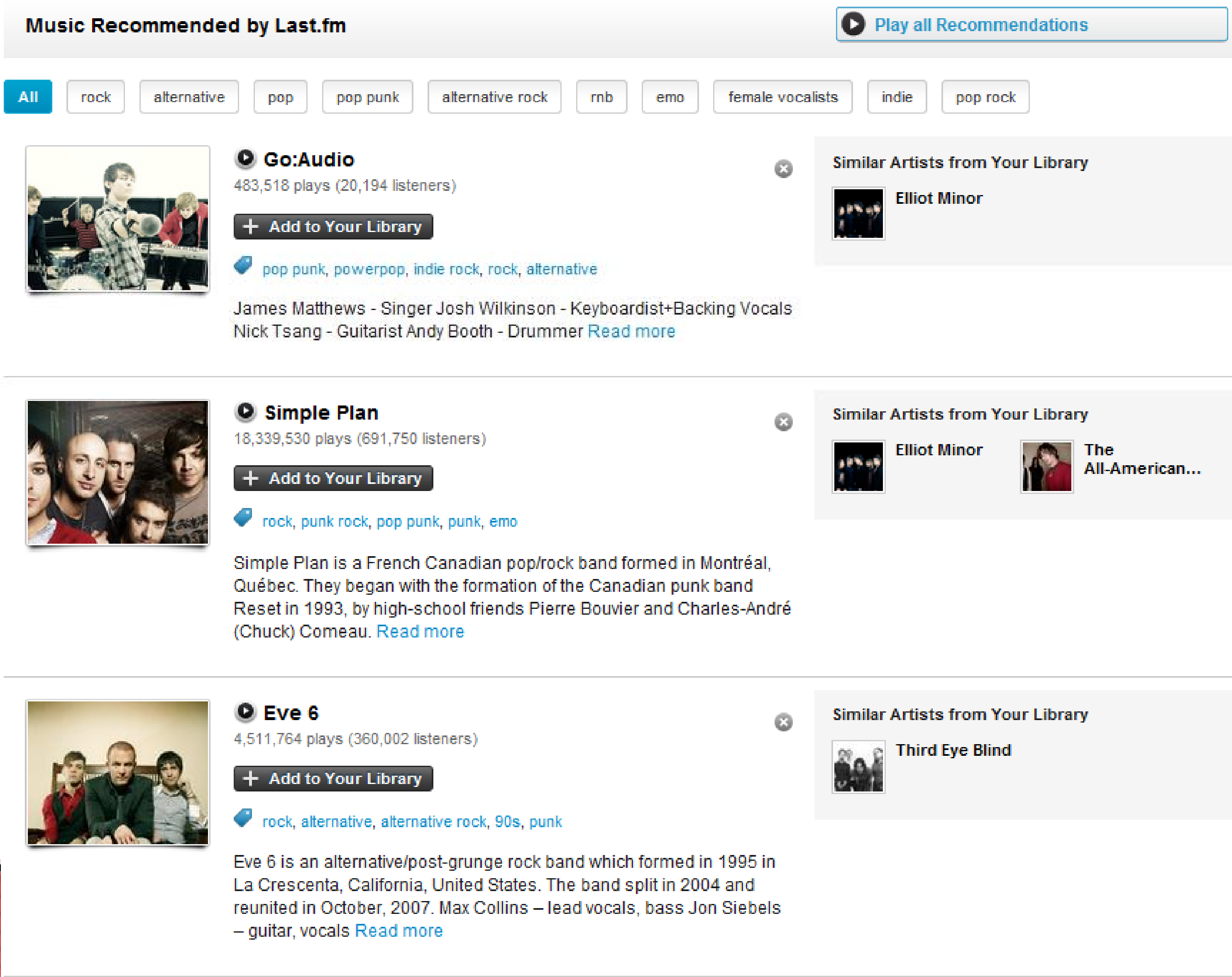
Outline

- The Network
- Collaborative Recommendation
- Collaborative Filtering
- The Social Graph
- Random Walks with Restarts
- Experiments and Results
- Conclusion

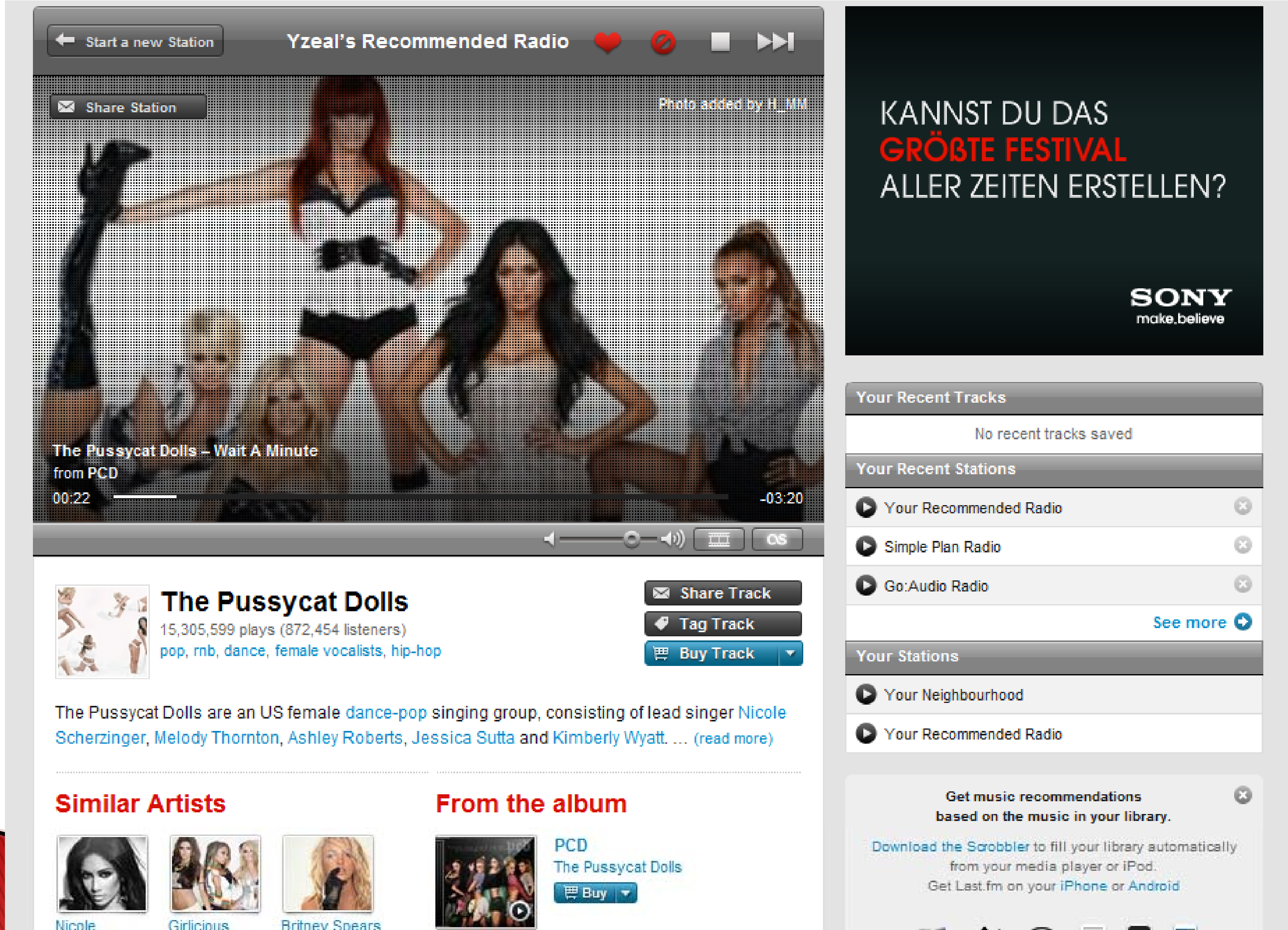
The Network: Last.fm



Last.fm – recommendations



Last.fm – recommendations (2)



Outline

- The Network
- Collaborative Recommendation
- Collaborative Filtering
- The Social Graph
- Random Walks with Restarts
- Experiments and Results
- Conclusion

Collaborative Recommendation

- Content-based
  - based on correlation between content and a user's preferences
  - limited to dictionary-bound relations
- Collaborative filtering (CF)

Outline

- The Network
- Collaborative Recommendation
- **Collaborative Filtering**
- The Social Graph
- Random Walks with Restarts
- Experiments and Results
- Conclusion

Collaborative Filtering (CF)

- **Memory-based**:
  - based on users‘ ratings of items
  - Aggregate ratings of k nearest neighbours.
- **Problems**:
  - integrating friendship and social tagging
  - choice for k

Collaborative Filtering (2)

- **Model-based**:
  - clusters based on similar rating behaviour
  - patterns recognised inside the clusters

Problems:
  - fine-tuning of parameters
  - generalisation of models
  - integrating additional information

Collaborative Filtering (3)

- User ratings are bounded and discrete (e.g. 0 to 5 stars on Amazon ).
- Memory-based CF systems:
  - user based
  - item based

CF – user based

- **User based:**
  - Predict rating $p_{a,i}$ based on users u with similar ratings to those of the active user a.
  - weighted combination of the ratings:

$$p_{a,i} = \boxed{\bar{r}_a} + \frac{\sum_{u=1}^{N} \boxed{(r_{u,i} - \bar{r}_u)}\,\boxed{w_{a,u}}}{\sum_{u=1}^{N} \boxed{w_{a,u}}}$$

a : active user
i : item not yet rated by a
$p_{a,i}$: predicted rating of a to i
$r_{u,i}$: rating of user u to i
$\bar{r}$: mean rating of user
$w_{a,u}$: similarity weight between a and u

CF – item based

- Item based:
  - Predict rating $p_{a,i}$ based on items k with similar ratings to item i.
  - weighted combination of the ratings:

$$p_{a,i} = \boxed{\bar{r}_i} + \frac{\sum_{k=1}^{M} \boxed{(r_{a,k} - \bar{r}_k)} \boxed{w_{i,k}}}{\sum_{k=1}^{M} \boxed{w_{i,k}}}$$

a	: active user
i	: item not yet rated by a
$p_{a,i}$	: predicted rating of a to i
$r_{a,k}$	: rating of items k by a
$\bar{r}$	: mean rating of item
$w_{i,k}$	: similarity weight between i and k

CF on Last.fm

- Item based systems:
  - significantly less items than users
  - true for most commercial applications

- Last.fm:
  - significantly more tracks than users
  - ⇨ Consider a user based system.

- important if we think about the sizes of the item–item or user–user similarity matrices

CF – Pearson correlation

- Compute the similarity weights $w_{a,u}$.
- Pearson correlation score is used:

$$w_{a,u} = \frac{\sum_{i=1}^{M} \boxed{(r_{a,i} - \bar{r}_a)} \boxed{(r_{u,i} - \bar{r}_u)}}{\boxed{\sigma_a \sigma_u}}$$

a	: active user
i	: item not yet rated by a
σ_a	: standard deviation of a's ratings
$r_{u,i}$	: rating of items i by user u
$\bar{r}$	: mean rating of item
$w_{i,k}$	: similarity weight between i and k

CF – integrating social components

- We rely on playcount instead of ratings.
- Integrate friendship and tagging into the model.
- ⇨ Compute three similarity weights based on
  - playcount,
  - tags and
  - friendships.

CF – integrating social components (2)

- Use the Pearson correlations coefficient for each.

- Replace the similarity weights with

$$w_{a,u} = \alpha w_{a,u}^{(I)} + \beta w_{a,u}^{(F)} + \gamma w_{a,u}^{(T)}$$

where

$$\alpha + \beta + \gamma = 1$$

and $w_{a,u}^{(I)}$, $w_{a,u}^{(F)}$ and $w_{a,u}^{(T)}$ are based on user tracks, user friendships and user tags respectively.

Different view on the problem

- View different similarities as a graph with different node types.

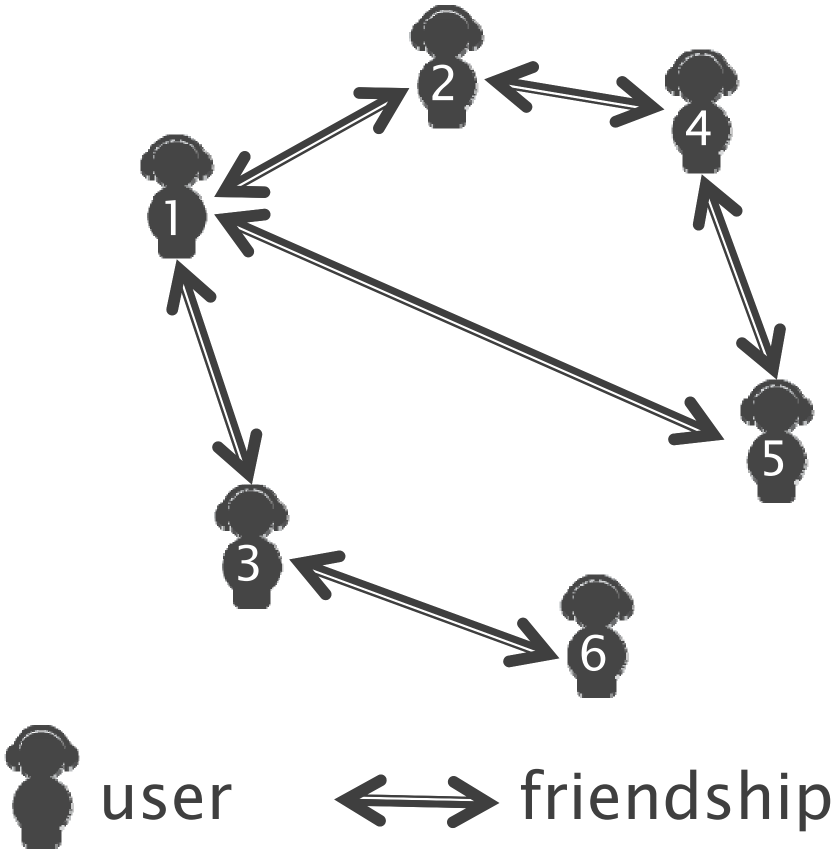
- Starting from a user: through friendship and preferences, where will he end up?

⇨ Random Walk model over the graph

Outline

- The Network
- Collaborative Recommendation
- Collaborative Filtering
- The Social Graph
- Random Walks with Restarts
- Experiments and Results
- Conclusion

Last.fm social graph – friends

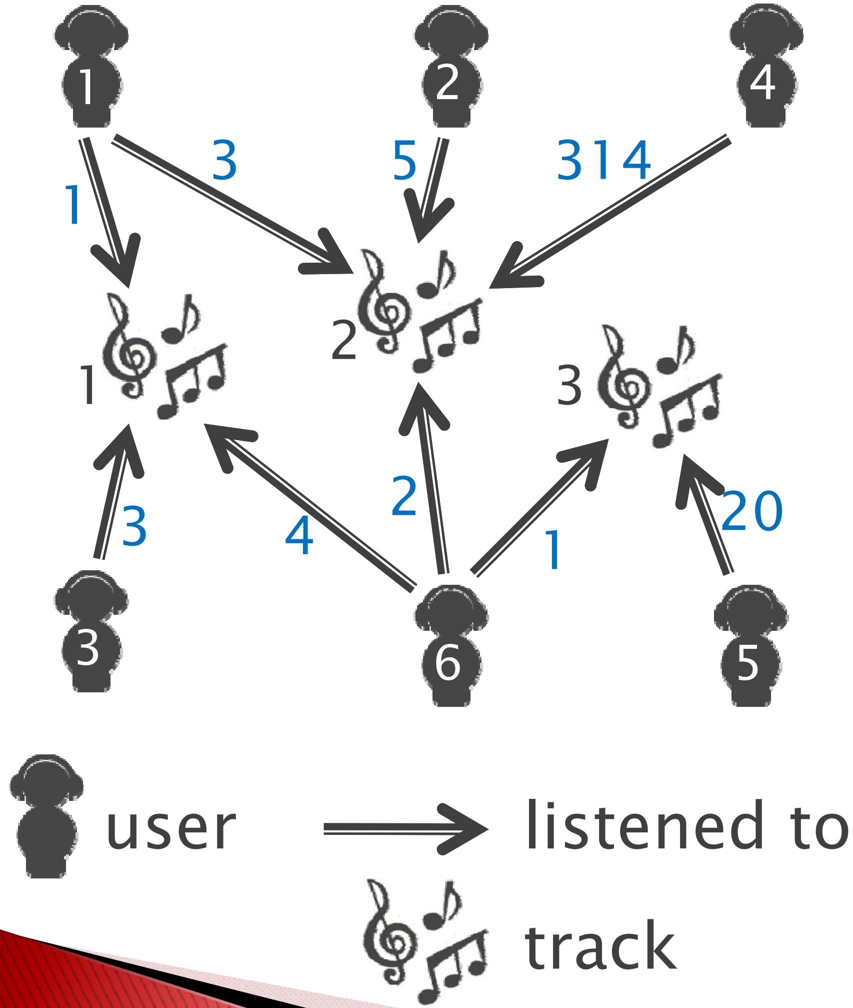


Corresponding adjacency matrix:

$$UU := \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

User–User matrix (**UU**)

Social graph – tracks



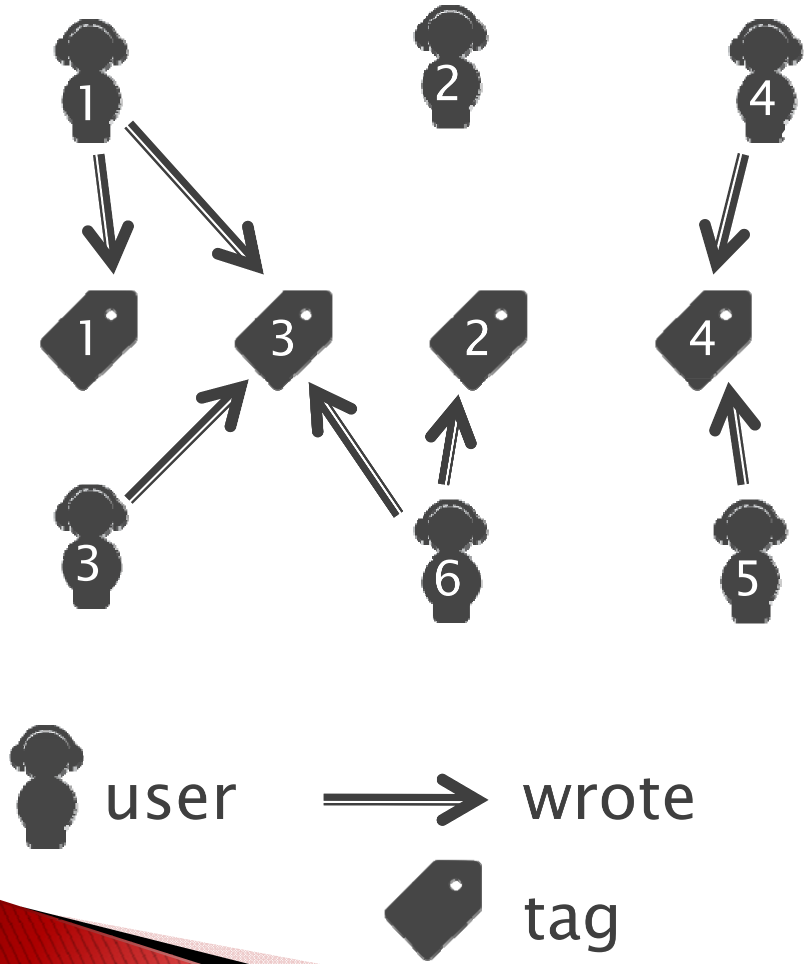
Corresponding adjacency matrix:

$$
\text{UTr} := \begin{bmatrix} 1 & 3 & 0 \\ 0 & 5 & 0 \\ 3 & 0 & 0 \\ 0 & 314 & 0 \\ 0 & 0 & 20 \\ 4 & 2 & 1 \end{bmatrix}
$$

User-Track matrix (**UTr**)

The numbers correspond to the times a user has listened to a track. They can be very high.

Social graph – tags

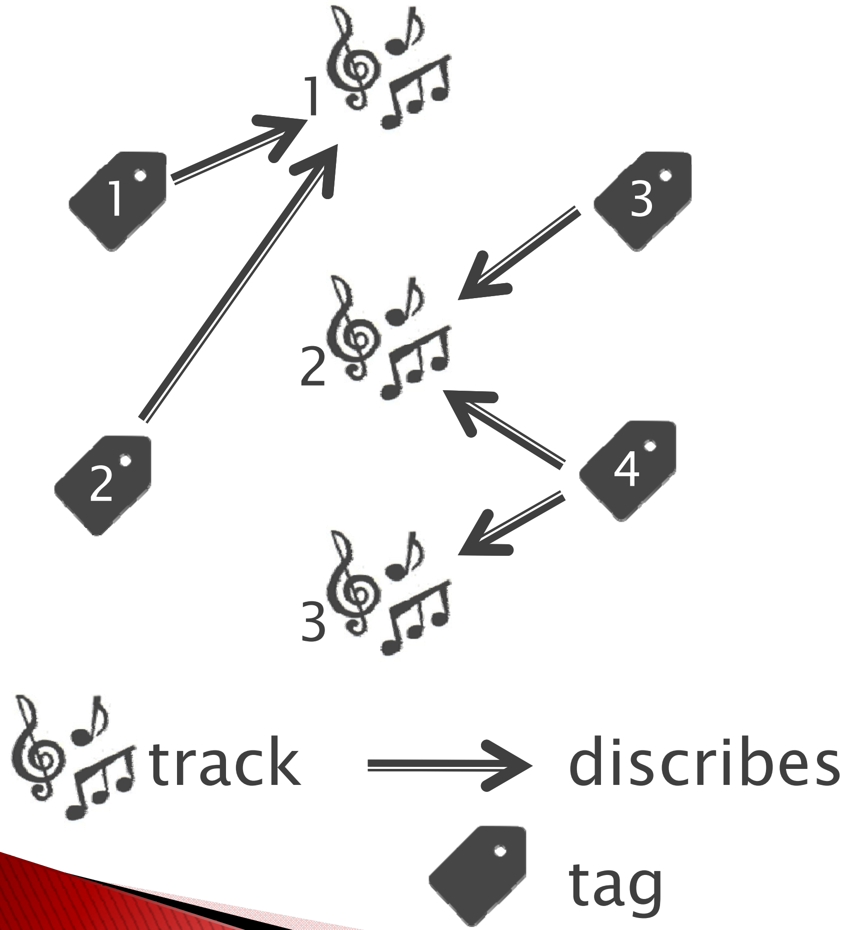


Corresponding adjacency matrix:

$$
UTg := \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}
$$

User-Tag matrix (**UTg**)

Social graph – tags/tracks

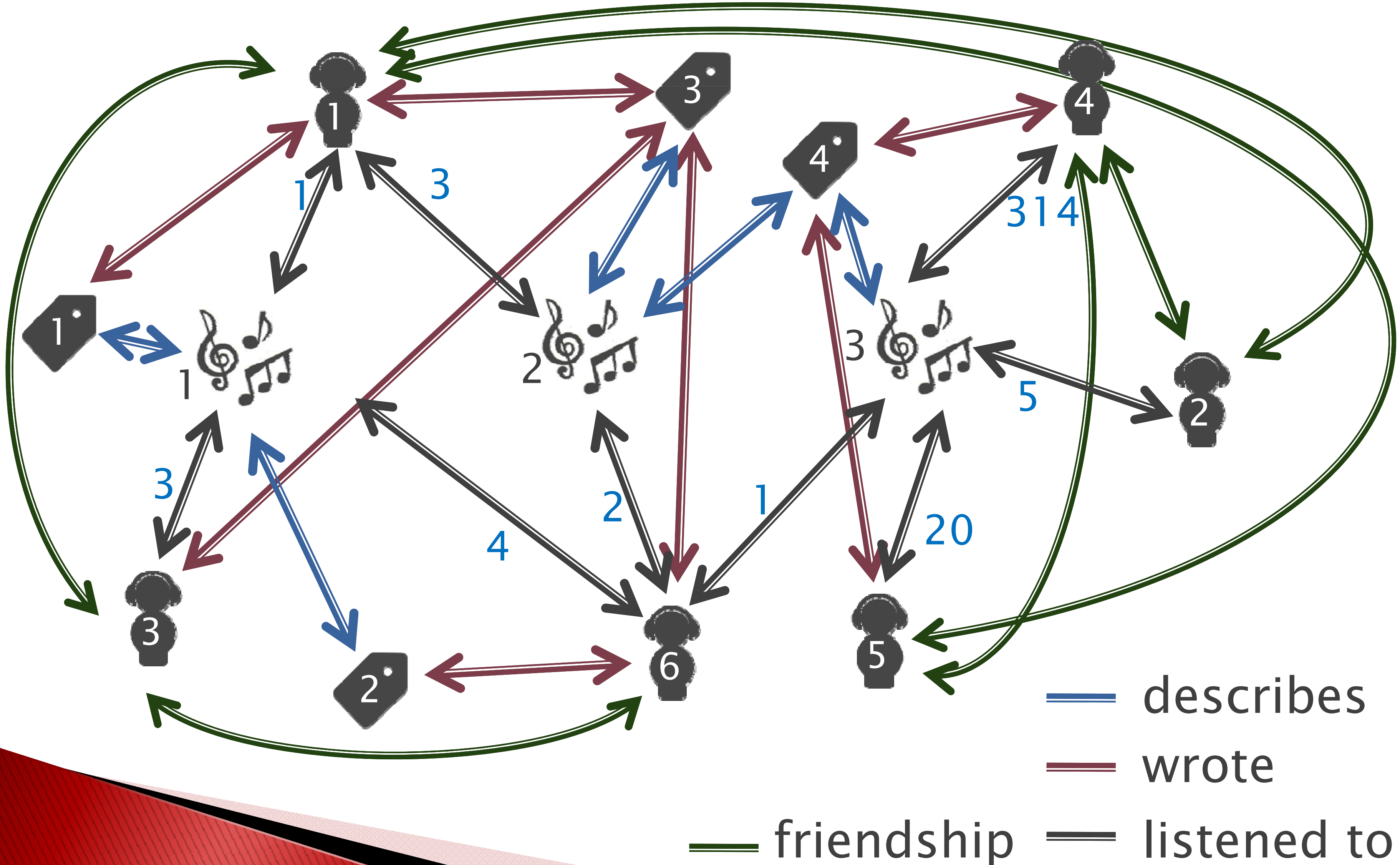


Corresponding adjacency matrix:

$$\text{TrTg} := \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Track–Tag matrix (**TrTg**)

Complete social graph



Complete social graph – matrix

What we call the social graph **S** is now made up of **UU**, **UTr**, **UTg** and **TrTg** as follows:

$$
\begin{array}{c|c|c|c|}
 & \text{Users} & \text{Tracks} & \text{Tags} \\
\hline
\text{Users} & UU & UTr & UTg \\
\hline
\text{Tracks} & UTr^{T} & 0 & TrTg \\
\hline
\text{Tags} & UTg^{T} & TrTg^{T} & 0 \\
\hline
\end{array}
=: S
$$

Social graph – normalisation

- playcount in **UTr** for each ranges from 1 to 11640
  ⇨ Normalise the other sub-matrices of **S**:

- Replace each bond of friendship in **UU** with the average user playcount.

- Apply an exponential decay function to the popularity values in **UTg** and **TrTg**.

- Most popular tags get the average user playcount.

Social graph – example

- In our example from before, where the average playcount is 39, that yields:

$$UU = \begin{bmatrix} 0 & 39 & 39 & 0 & 39 & 0 \\ 39 & 0 & 0 & 39 & 0 & 0 \\ 39 & 0 & 0 & 0 & 0 & 39 \\ 0 & 39 & 0 & 0 & 39 & 0 \\ 39 & 0 & 0 & 39 & 0 & 0 \\ 0 & 0 & 39 & 0 & 0 & 0 \end{bmatrix} \quad UTg = \begin{bmatrix} 14 & 0 & 14 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 14 & 0 \\ 0 & 0 & 0 & 39 \\ 0 & 0 & 0 & 39 \\ 0 & 14 & 14 & 0 \end{bmatrix} \quad TrTg = \begin{bmatrix} 14 & 14 & 0 & 0 \\ 0 & 0 & 14 & 39 \\ 0 & 0 & 0 & 39 \end{bmatrix}$$

Here I simply assumed $f(x) = 39 * e^{-x}$ as the exponential decay function and truncated the values.

Social graph – example (2)

- We want to use S as a transition probability table.
- ⇨ Normalise S.

$$UU = \begin{bmatrix} 0 & 39 & 39 & 0 & 39 & 0 \\ 39 & 0 & 0 & 39 & 0 & 0 \\ 39 & 0 & 0 & 0 & 0 & 39 \\ 0 & 39 & 0 & 0 & 39 & 0 \\ 39 & 0 & 0 & 39 & 0 & 0 \\ 0 & 0 & 39 & 0 & 0 & 0 \end{bmatrix} \quad UU' = \begin{bmatrix} 0 & \frac{39}{117} & \frac{39}{117} & 0 & \frac{39}{117} & 0 \\ \frac{39}{78} & 0 & 0 & \frac{39}{78} & 0 & 0 \\ \frac{39}{78} & 0 & 0 & 0 & 0 & \frac{39}{78} \\ 0 & \frac{39}{78} & 0 & 0 & \frac{39}{78} & 0 \\ \frac{39}{78} & 0 & 0 & \frac{39}{78} & 0 & 0 \\ 0 & 0 & \frac{39}{39} & 0 & 0 & 0 \end{bmatrix} \quad UU' = \begin{bmatrix} 0 & \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

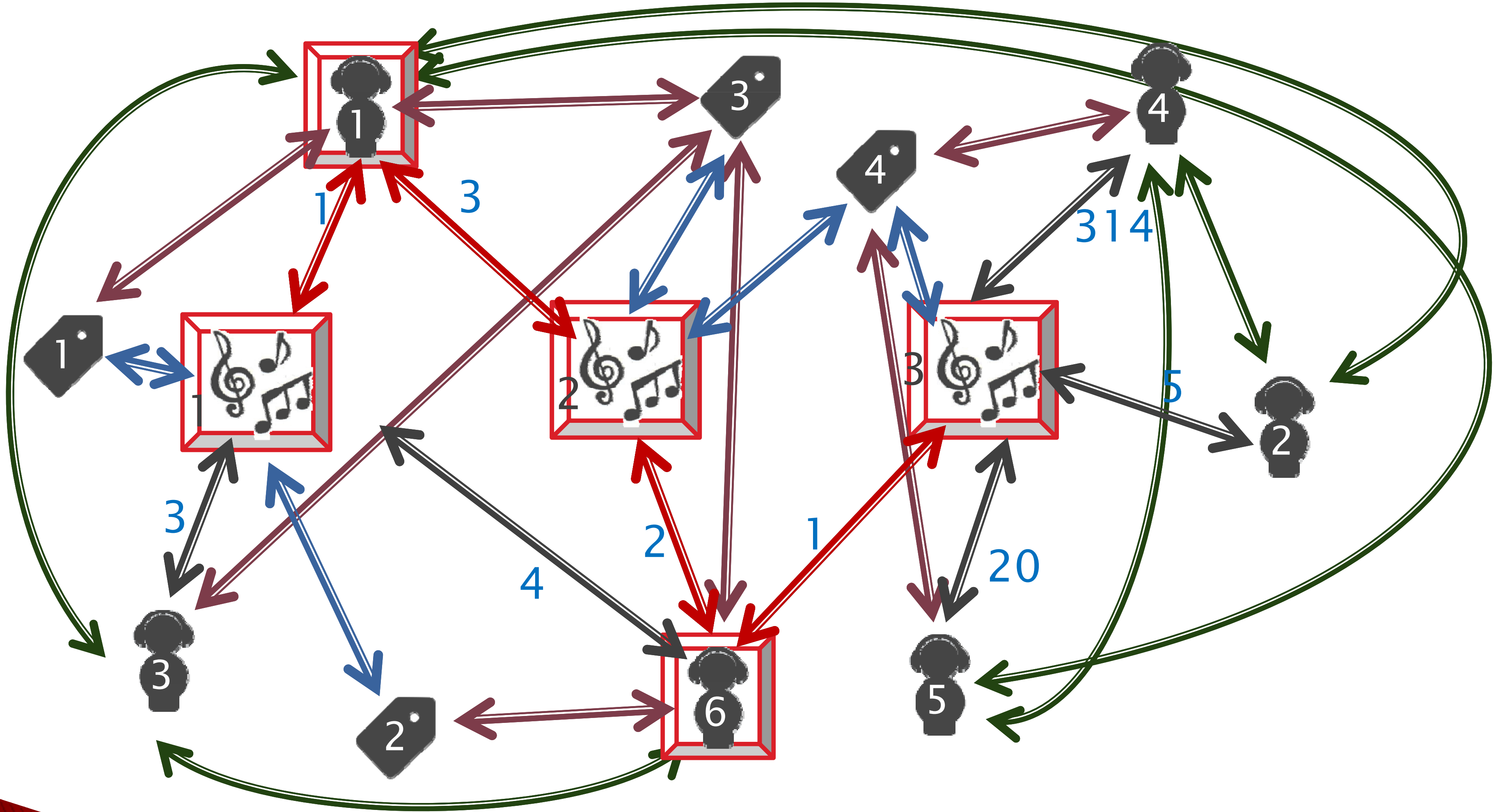
Outline

- The Network
- Collaborative Recommendation
- Collaborative Filtering
- The Social Graph
- **Random Walks with Restarts**
- Experiments and Results
- Conclusion

Random Walks with Restarts (RWR)

- We use Random Walks to compute the relatedness of two nodes in a graph.
- Start with node x.
- Perform a Random Walk by randomly following links to other nodes.
- There's also a probability a to restart at x in each step.

A Random Walk



RWR – definitions

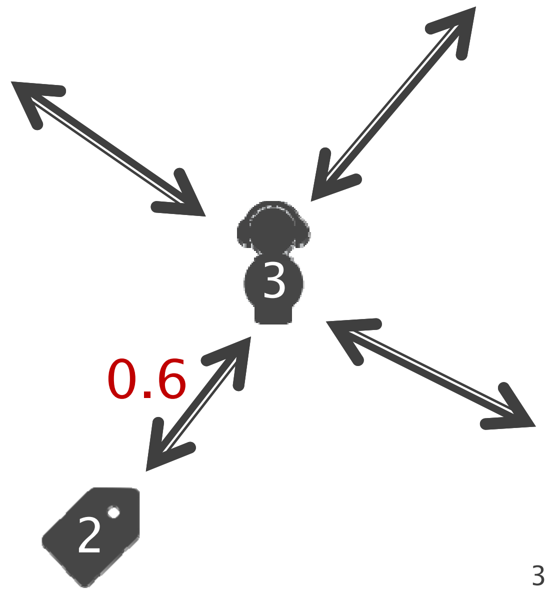
- $\mathbf{p}^{(t)}$ is a column vector.
  $\mathbf{p}_i^{(t)}$ is probability that we are at node i in step t

- q corresponds to the starting setup.

- S (column normalised) is the transition probability table. Its elements $S_{i,j}$ give the probability of j being the next state if we are in state i.

$$
\begin{array}{cc}
 & \begin{array}{cccc} & j=2 & & \end{array} \\
\begin{array}{c} \\ \\ i=3 \\ \\ \end{array} & \begin{bmatrix} 0.3 & 0.5 & 0.1 & 0.1 \\ 0.4 & 0.1 & 0.2 & 0.3 \\ 0.1 & \boxed{0.6} & 0.1 & 0.2 \\ 0.1 & 0.1 & 0.1 & 0.7 \end{bmatrix}
\end{array}
$$

S



RWR – probabilities

- steady-state probabilities for each node: Apply

$$\mathbf{p}^{(t+1)} = (1-\alpha)\mathbf{S}\mathbf{p}^{(t)} + \alpha\mathbf{q}$$

until convergence.

- ⇨ long-term visit rates of each node based on the starting node x.

RWR – relatedness

- $\mathbf{p}_i^{(l)}$ is the measure of relatedness between x and i.

- By taking into account users'
  - music taste,
  - tagging behaviour and
  - friendships

  the Random Walks method allows us to predict preferences of users to paricular tracks.

Outline

- The Network
- Collaborative Recommendation
- Collaborative Filtering
- The Social Graph
- Random Walks with Restarts
- **Experiments and Results**
- Conclusion

Experiments: Last.fm – data

- The following amount of data was extracted from the Last.fm social network:

- 3148 users
- 30520 tracks
- 12565 tags
- 5616 bonds of friendship

- In order to make the resulting matrices less sparse, only very active users were selected. The rest of the dataset was populated with respect to those users.

Experiments – ground truth

- 20% of the tracks each user has listened to will be set to 0 in UTr.
- We can later compare our results to this ground truth.

Experiments – CF

- Predict the playcount of the tracks in S for each user.

  ⇨ ranked vector of the best 1000 tracks in descending order of playcount

- Playcount variables have a wide range (1–11640) .

  ⇨ Standardise the playcount and the mean of the nearest neighbours.

Experiments – CF (2)

- Substract the mean and devide with the standard deviation:

$$p_{a,i} = \frac{\sum_{u=1}^{n} \left(\boxed{\frac{r_{u,i} - \bar{r}_a}{\sigma_a}} - \boxed{\frac{\bar{r}_u - \bar{r}_a}{\sigma_a}} \right) \boxed{w_{a,u}}}{\boxed{\sum_{u=1}^{n} w_{a,u}}}$$

σ_a : standard deviation of a's ratings
a : active user
i : item not yet rated by a
$p_{a,i}$: predicted rating of a to i
$r_{u,i}$: rating of user u to i
$\bar{r}$: mean rating of user
$w_{a,u}$: similarity weight between a and u

Experiments – CF – parameters

- The CF method adopted in these experiments uses two parameters:

  - The treshold number of common tracks between users Tr, emperically estimated to 20.

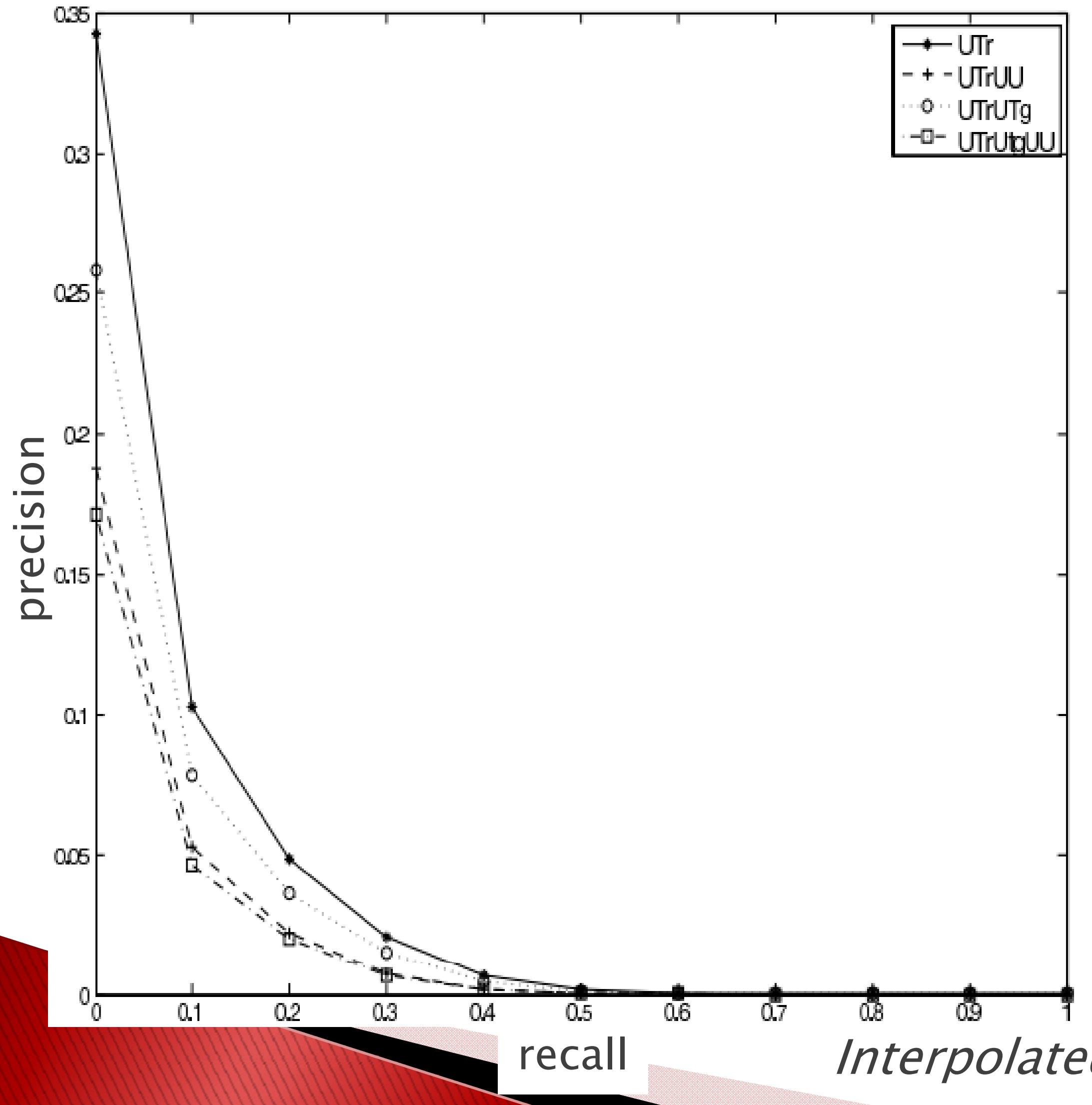
  - The number of nearest neighbours *k* with the highest correlation to a user, again estimated to 15.

Both were estimated using a withheld part of the graph.

Experiments – CF – matrices

- Three user-user similarity matrices were computed:
  - $w_{a,u}^{(I)}$: based on **UTr**, the standard used in CF systems
  - $w_{a,u}^{(F)}$: based on **UU**, measuring correlation based on friendship
  - $w_{a,u}^{(T)}$: based on **UTg**, measuring correlation based on collaborative annotation

Experiments – CF – results



Interpolated Recall-Precision Curves

Based on:

UTr: tracks

UTrUU: tracks and friendships

UTrUTg: tracks and tags

UTrUTgUU: tracks, tags and friendship

Experiments – CF – results (2)

- Additional information about friendship and social tagging makes the performance worse.

- Possible reason:

  Weights and parameters, have not been tuned finely enough.

- So the memory based CF method cannot incorporate social knowledge in a trivial way.

Experiments – RWR

- Top 1000 tracks in descending order for each user:

- N: numer of columns in **S**
- u_a : user currently under evaluation

- Create a query vector **q** with $q_i = 1$ if $S_{u,i} > 0$ and $q_{u_a} = 1$, $i=1..N$.

Experiments – RWR (2)

- Normalise **q** so that $\|\mathbf{q}\|_1 = 1$.

- A series of experiments was performed to determine the restart probability $a = 0.8$.

- high restart probability a
  ⇨ Model goes back to the initial **q** more frequently.
    ⇨ It stays in the neighbourhood of u_a.

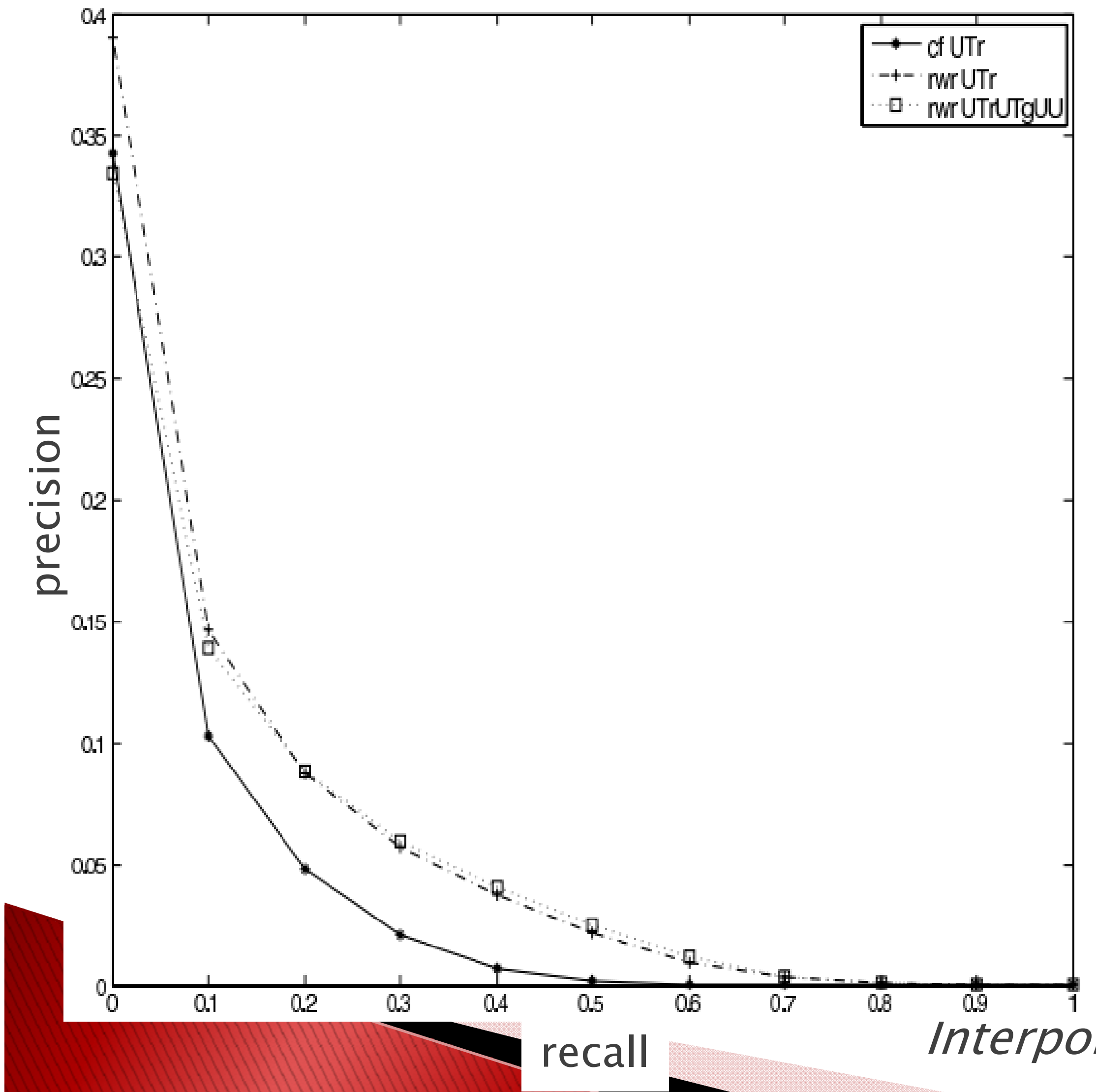
Experiments – RWR (3)

- Perform a Random Walk on S and get the
  ⇨ stationary probability vector for u_a.

- Order the tracks in descending order and select the first 1000.

- Compare them to the ground truth.

Experiments – RWR – results



Interpolated Recall-Precision Curves

cf UTr:
CF method based on tracks

rwr UTr:
Random Walks based on tracks

rwr UTrUTgUU:
Random Walks based on the whole social graph S

Experiments – results

- Random Walks outperforms the baseline CF method.
- Random Walks using **S** is not significantly better than Random Walks using only **UTr**.

Experiments – results – numbers

- Let‘s take a closer look at the numbers:

		P5	P20	P1000	num_rel_ret
cf	UTr	0.1472	0.0934	0.0144	45268
	UTrUU	0.0719	0.0470	0.0104	32862
	UTrUTg	0.1046	0.0698	0.0129	40546
	UTrUTgUU	0.0628	0.0416	0.0102	32106
rwr	UTr	**0.1747**	**0.1229**	**0.0221**	**69514**
	UTrUU	**0.1726**	**0.1229**	**0.0222**	**69742**
	UTrUTg	0.1486	**0.1145**	**0.0227***	**71404***
	UTrUTgUU	0.1483	**0.1139**	**0.0228**	**71645**

- **Bold** numbers indicate statistical significance (at $p < 0.001$).

Experiments – results – numbers (2)

		P5	P20	P1000	num_rel_ret
cf	UTr	0.1472	0.0934	0.0144	45268
	UTrUU	0.0719	0.0470	0.0104	32862
	UTrUTg	0.1046	0.0698	0.0129	40546
	UTrUTgUU	0.0628	0.0416	0.0102	32106
rwr	UTr	**0.1747**	**0.1229**	**0.0221**	**69514**
	UTrUU	**0.1726**	**0.1229**	**0.0222**	**69742**
	UTrUTg	0.1486	**0.1145**	**0.0227***	**71404***
	UTrUTgUU	0.1483	**0.1139**	**0.0228**	**71645**

- The number of relevant retrieved (num_rel_ret) tracks using **UTrUTg** is significantly higher than that using **UTrUU.**

Experiments – results

- In Last.fm users usually spend
  - more time looking for music they like and annotating
  - than looking for friends.

⇨ UU matrix is relatively sparse compared to the UTg matrix.

⇨ It doesn‘t influence the results as significantly.

- This is typical for special interest social networks like Last.fm, Flickr or YouTube (unlike Facebook).

Experiments – results – reasons

- Random Walks outperforms CF because
  - the graph model captures relationships better
  - we can recognise more elaborate patterns and rules.

Outline

- The Network
- Collaborative Recommendation
- Collaborative Filtering
- The Social Graph
- Random Walks with Restarts
- Experiments and Results
- **Conclusion**

Conclusion

- Random Walk with Restarts doesn‘t need so much fine-tuning of parameters
  ⇨ it is more robust than CF.

- lack of scalability:
  - On-line response time is not acceptable when applied to large social graphs in situations that need real-time response, like searching.

⇨ Use an approximation of RWR instead (with >90% quality).

Thank you for your attention!

Resources

- Ioannis Konstas, Vassilios Stathopoulos, Joemon M. Jose: On social networks and collaborative recommendation, SIGIR Conference 2009, pp. 195-202.