Collaborative Information Filtering for the Internet

Application and Evaluation

Ye Huang, Jie Ren, Tao Zhou, Zike Zhang,
Matúš Medo, Yi-Cheng Zhang

Université de Fribourg

COST P10 “Physics of Risk”
2nd Internal Workshop, June 13, 2007

Our goal

- every user collects interesting webpages
- we want to promote nice unnoticed pieces to users

Our goal

- every user collects interesting webpages
- we want to promote nice unnoticed pieces to users
- meanwhile:
  - the number of collected webpages is large
  - the number of users is very large

Our goal

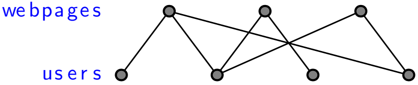
- every user collects interesting webpages
- we want to promote nice unnoticed pieces to users
- meanwhile:
  - the number of collected webpages is large
  - the number of users is very large
- to handle the data we need an effective prediction algorithm
- available algorithms:
  - **Global Ranking Method**
    (recommendation by the popularity)
  - **Collaborative Filtering**
    (recommendation by the user-user similarity)

Modifications of the Quantum Diffusion

- a collection of webpages is a binary system (we either collect or not)

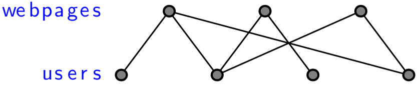
Modifications of the Quantum Diffusion

- a collection of webpages is a binary system
  (we either collect or not)
- all information about the system: unweighted bipartite graph



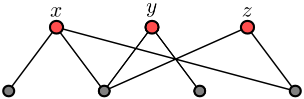
Modifications of the Quantum Diffusion

- a collection of webpages is a binary system
  (we either collect or not)
- all information about the system: unweighted bipartite graph

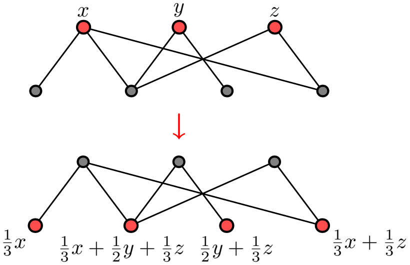


- good news: one iteration step is enough
  - saves computer memory
  - saves CPU time

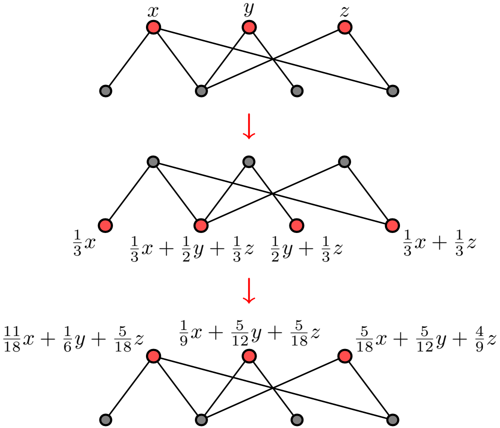
The starting point – resources redistribution



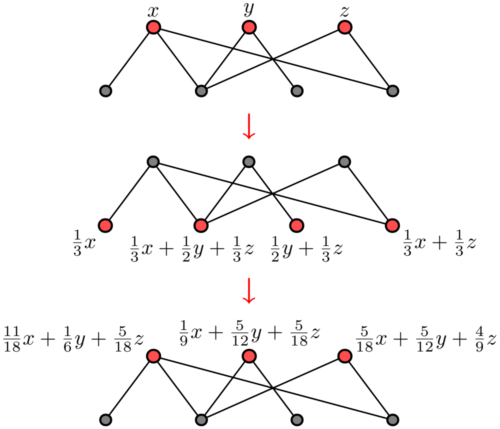
The starting point – resources redistribution



The starting point – resources redistribution



The starting point – resources redistribution



$$R'_{\alpha} = \sum_{\beta} W_{\alpha\beta} R_{\beta}$$

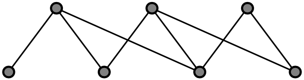
How to make a recommendation?

- we make a recommendation for every user separately
- we label initial resources as R_α ($\alpha = 1, \ldots, N$)
- this initial condition is fixed for every user separately:
  - if user i has collected webpage α, we set $R_\alpha = 1$
  - otherwise $R_\alpha = 0$

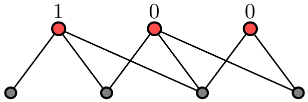
How to make a recommendation?

- we make a recommendation for every user separately
- we label initial resources as R_α ($\alpha = 1, \dots, N$)
- this initial condition is fixed for every user separately:
  - if user i has collected webpage α, we set $R_\alpha = 1$
  - otherwise $R_\alpha = 0$
- the final resources can be written as $R'_\alpha = W_{\alpha\beta} R_\beta$
  - here the matrix W is given by the two-step flow process we have shown before
  - in this way **we propagate user's opinions** over the object-object network

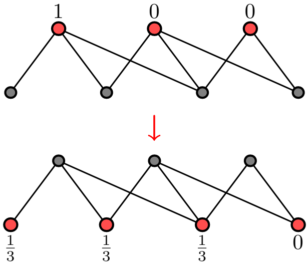
How it works in practice



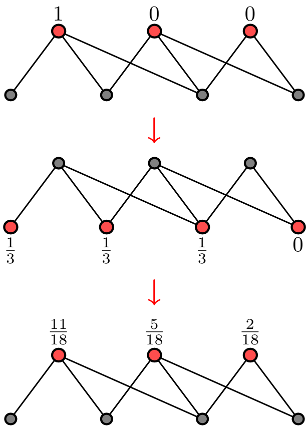
How it works in practice



How it works in practice



How it works in practice



Numerical tests of the method

- we use MovieLens data:
  - 943 users, 1682 movies (`www.grouplens.org`)
  - integer ratings 1,2,3,4,5
- "coarse graining" used to remove weights from the edges:
  - if for user i and movie α is $v_{i\alpha} \geq 3$, we draw the link i — α
- finally we have 85 250 edges (5% of the full graph)

Numerical tests of the method

- we use MovieLens data:
  - 943 users, 1682 movies (`www.grouplens.org`)
  - integer ratings 1,2,3,4,5
- "coarse graining" used to remove weights from the edges:
  - if for user i and movie α is $v_{i\alpha} \geq 3$, we draw the link $i — \alpha$
- finally we have 85 250 edges (5% of the full graph)
- we transfer 10% of the data to the probe
  - appreciated but hidden to the algorithm
- the remaining 90% is used for the recommendation
  - used to obtain recommendations

A measure of the method's performance

- for user i we obtain a sorted list of uncollected objects
- let's assume that:
  - user 1 has together 10 objects in the probe
  - among the first 20 objects recommended by the algorithm there are 6 probe objects

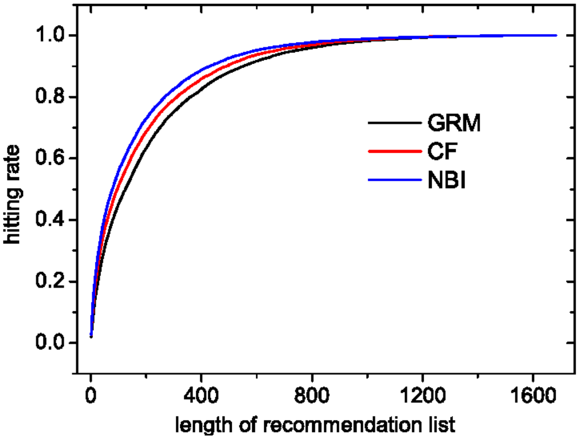
A measure of the method's performance

- for user i we obtain a sorted list of uncollected objects
- let's assume that:
  - user 1 has together 10 objects in the probe
  - among the first 20 objects recommended by the algorithm there are 6 probe objects
- the hitting rate for user 1 and the list length $L = 20$ is

$$h_1(L) := \frac{\text{recovered probe entries}}{\text{total probe entries}} = \frac{6}{10} = 0.6$$

  - the denominator is fixed and the numerator grows with L
- by averaging over all users we obtain $h(L)$

Comparison of various methods



Conclusion

- we have a promising tool in our hands
  - smaller computational complexity and better results than correlation-based methods
  - no careful fine-tuning needed
- things to be done:
  - to test the method in detail
  - to implement the method with a really bipartite data (no coarse graining)

- thank You for Your attention
- thanks to prof. Frank Schweitzer for leading the COST-SWITZERLAND project
- thanks to all members of our group