

Internet Analytics (COM-308): Midterm Exam

April 13, 2015

Duration: **2h00**.

Total points: **100**. There is a bonus question for additional points; we suggest you do this last.

Number of pages: **11**.

Allowed documents: **class notes, lab handouts, homeworks, your own code**.

There should in general be enough room below every question for intermediate calculations and your answer. However, you are allowed to use additional sheets of paper; please **write your name on every sheet**, number them, and staple them to this document before handing in.

The use of **mobile phones, tablets, laptop computers**, and other communication devices is **prohibited**.

Name:
First name:
SCIPER number:
Signature:

Please leave blank.

1	2	3	4	5	Total
20	15	20	25	20	100

Question 1: Graph Conductance and Random Walk (20+5* points)

Consider the unrooted complete binary tree G of n nodes: all the non-leaf nodes have degree 3 (See Fig. 1 for an example with $n = 22$).

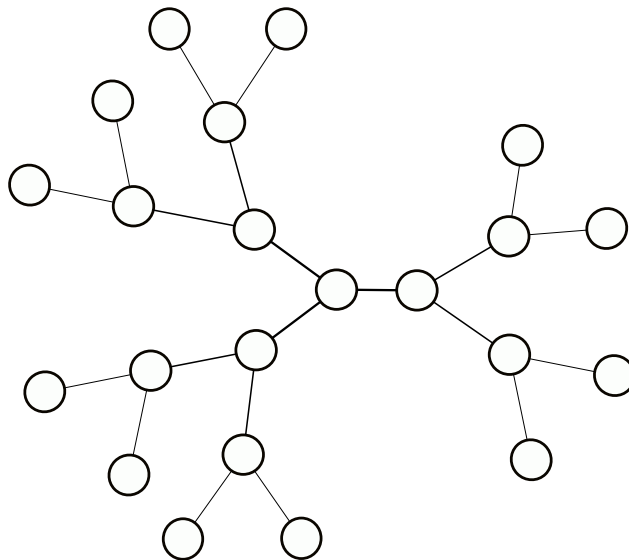


Figure 1: Complete unrooted binary tree G with $n = 22$.

1. **Stationary distribution (7 pts)** Find the stationary distribution of the random walk $\{X_t\}$ on G as a function of n . (To ensure that the random walk is ergodic, you may assume that there is at least one state with a self-transition of very small probability; you can ignore this in the calculation).

2. **Conductance (8 pts)** Compute the conductance $\Phi(G)$ of the graph G as a function of n .

3. **Mixing rate (5 pts)** Find a time bound $t(n)$ for n large that guarantees that all the transition probabilities $p_{ij}(t)$ of the random walk are within ϵ of the stationary distribution probabilities π_j , i.e. $\|p_{ij}(t) - \pi_j\| \leq \epsilon$. Hint: You can use the approximation $(1 - \alpha(n))^t \sim 1 - \alpha(n)t$ when $\alpha(n)$ small and t large.

4*. **Bonus (5 pts)** A chain graph Ch_n of length n has exactly the same numbers of nodes and edges as the binary tree described above. Compute the conductance of the graph Ch_n and give an intuitive justification why it is lower/higher/equal than the one for the described tree.

Question 2: Network Structure (15 points)

1. **Clustering (6 pts)** We saw in class that the expected clustering coefficient of the $G(n, p)$ random graph is p . Let us compare the clustering coefficient of the graphs given below with that of a $G(n, p)$ with the same (expected) number of edges. For each case, answer higher/lower/equal if the graph has a higher/lower/equal clustering coefficient compared to the $G(n, p)$. Give a short justification.

- The n -cycle C_n .

☐ higher

☐ lower

☐ equal

- The complete graph K_n .

☐ higher

☐ lower

☐ equal

- The 2-dimensional square lattice (grid).

☐ higher

☐ lower

☐ equal

- The Watts-Strogatz graph with $k = 3$ and no shortcuts.

☐ higher

☐ lower

☐ equal

2. **Friendship paradox (9 pts)** We computed in class the average number of friends μ , and the average number of friends' friends $\mu(1 + \sigma^2/\mu^2)$, where σ^2 is the empirical degree variance over all the nodes. We are interested in the ratio $(1 + \sigma^2/\mu^2)$ between the two averages. Compute this ratio for the following graphs:

- A star network with n leaf nodes (of degree $d_v = 1$) and a central node (of degree $d_v = n$).

- The $G(n, p)$ random graph, with $p = c/n$ and $c > 0$ a constant. Hint: you may assume that all degrees d_v are i.i.d.; $\text{Var}(\text{Binom}(n, p)) = np(1 - p)$.

Question 3: Computing a Gradient with MapReduce (20 points)

In the lecture, we said that latent factor models used for recommender systems can be trained using gradient descent. In this question, we ask you to give a parallel implementation of the gradient computation using MapReduce.

Recall that we model the ratings matrix R of size $n \times m$ (for n users and m movies) as a rank K matrix, i.e. $R \approx P^T Q$. We define the set of all movies rated by some user u as R_u . Omitting regularization, we write the gradient with respect to a single user factor $p_u \in \mathbb{R}^K$, when all item factors Q are given, as:

$$\nabla_{p_u} E(P, Q) = -2 \sum_{i \in R_u} (r_{ui} - p_u^T q_i) q_i$$

Note that each summand corresponds to a rating, and contributes only to the corresponding ∇_{p_u} . The complete gradient $\nabla_P E(P, Q)$ then consists of all the $\nabla_{p_u} E(P, Q)$, i.e.

$$\nabla_P E(P, Q) = \{\nabla_{p_u} E(P, Q) \mid u \in \text{users}\}$$

Assume that you have access to the global functions

- `Vector getItemFactor(int i)`, which returns the item factor q_i , and
- `Vector getUserFactor(int u)`, which returns the user factor p_u .

The goal is to write a MapReduce job that computes the elements of the set $\{\nabla_{p_u} E(P, Q) \mid u \in \text{users}\}$.

1. **(1 pt)** What is the dimension of $\nabla_{p_u} E(P, Q)$?
2. **Mapper (6 pts)** The mapper receives as input a `NullWritable` key (i.e. no key) and a `Vector` value containing $[u, i, r_{ui}]$ for all $(u, i) \in R$. Write (in pseudo-code) the implementation of the mapper. Give the type and describe the meaning of the output key. Give the type and describe the meaning of the output value.

3. **Reducer (6 pts)** Write (in pseudo-code) the implementation of the reducer. Give the type and describe the meaning of the output key. Give the type and describe the meaning of the output value.

4. **Unit Test (7 pts)** To test your code, we provide, as usual, a unit test. There are two users and three items. The dataset consists of the following four ratings:

u	i	r_{ui}
1	1	3
1	3	1
2	1	2
2	2	1

The values of the user and item factors are:

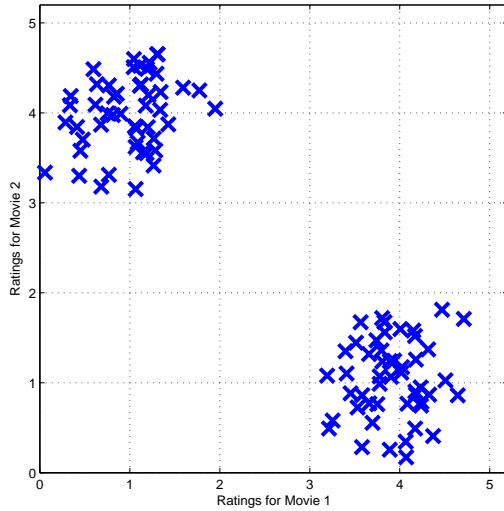
$$p_1 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} \quad p_2 = \begin{bmatrix} -0.1 \\ 0.3 \end{bmatrix} \quad q_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad q_2 = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \quad q_3 = \begin{bmatrix} 2 \\ -2 \end{bmatrix}$$

- (a) Compute the output of the Mapper.

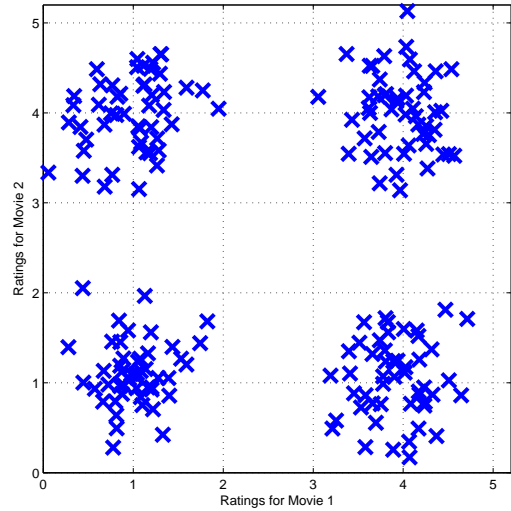
- (b) Compute the output of the Reducer.

Question 4: A Tale of Two Movies (25 points)

Part 1. Consider a movie recommendation task. We collected two different datasets, A and B. Both consist of ratings of users for two movies. We plot these datasets in Figures 2a and 2b. There is a point (x, y) for each user. The x coordinate is the rating of the corresponding user for Movie 1 and the y coordinate the rating for Movie 2.



(a) Dataset A



(b) Dataset B

Figure 2: Ratings of two movies. Each data point represents the ratings of a user for both movies.

The vectors v_1 and v_2 in Equation (1) are the principal components of Dataset A, with some coordinates missing.

$$v_1 = \begin{pmatrix} -\frac{1}{\sqrt{2}} \\ \boxed{} \end{pmatrix} \quad v_2 = \begin{pmatrix} \boxed{} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \quad (1)$$

- (2 pt) What two properties do the vectors v_1 and v_2 have to satisfy?
- (3 pt) Fill in the missing values in Equation (1) and sketch the vectors into Figure 2a (don't forget to label them).
- (3 pt) How much variance can be retained by projecting onto the first principal component v_1 ?
 - ☐ At least 75%
 - ☐ At least 25%, but less than 75%
 - ☐ Less than 25%
- (2 pt) If you knew that a user rated Movie 1 as 5, would you recommend Movie 2 to that user? Justify your answer.

We now focus on Dataset B, shown in Figure 2b. Assume that you are given principal components u_1 and u_2 for Dataset B.

5. **(3 pt)** How much variance can be retained by projecting onto the first principal component u_1 ?
 - ☐ At least 75%
 - ☐ At least 25%, but less than 75%
 - ☐ Less than 25%
6. **(2 pt)** If you knew that a user rated Movie 1 as 5, would you recommend Movie 2 to that user? Justify your answer.

Part 2. Consider the following matrices. In the questions below, we index rows by u and columns by i .

$$\begin{matrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} & \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \\ (a) & (b) & (c) & (d) \end{matrix} \quad (2)$$

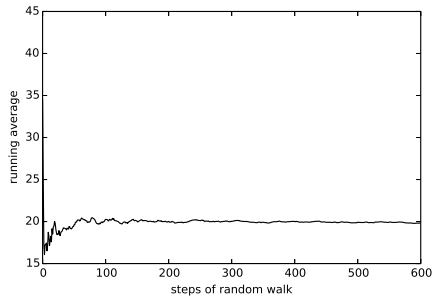
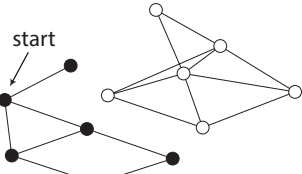
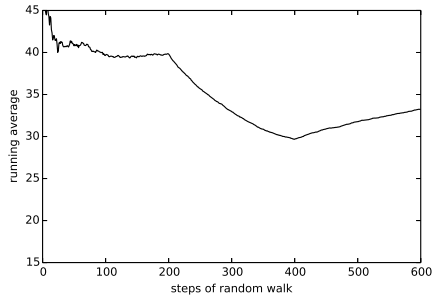
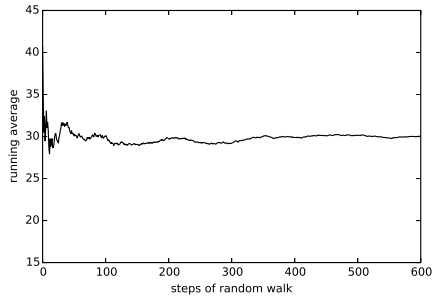
1. **(5 pt)** Which of the matrices in Equation (2) can be represented exactly by a one-dimensional factor model $\hat{r}_{ui} = p_u \cdot q_i$ (that is, $p_u, q_i \in \mathbb{R}$)? If you can represent a matrix, provide the factors. If you cannot, argue why not.
2. **(5 pt)** Now consider the baseline predictor $\hat{r}_{ui} = b_u + b_i$, i.e. per-user and per-item constants which are added together. Which of the matrices in Equation (2) *cannot* be represented exactly by the baseline predictor? If you can represent a matrix, provide the parameters b_u and b_i .

Question 5: Walking on a Social Network (20 points)

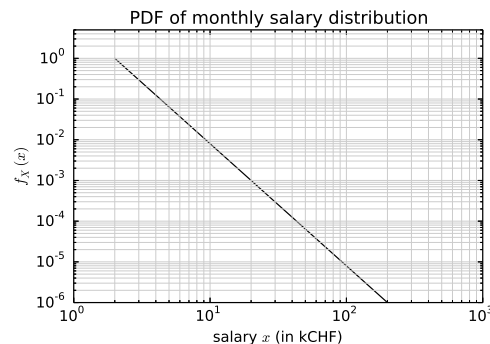
Part 1. (6 pts) The plots below show the running (unbiased) estimate of average age, obtained with a random walk on different social networks. We know that each network contains *old* nodes, whose average age is around 40, and *young* nodes, whose average age is around 20. In each case

1. draw a plausible network that could give rise to such a running estimate,
2. indicate young nodes in dark and point to a plausible starting point for the walk, and
3. give a short explanation, including key network properties.

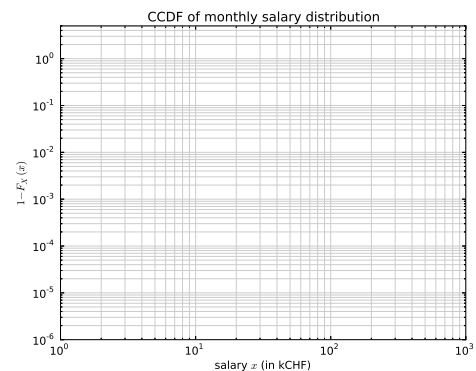
The first row contains an example.

realization of random walk	sketch of network	explanations
		<p>Apparently, we are stuck on young nodes. This might be because the network is composed of two completely disconnected components.</p>
		
		

Part 2. Suppose that you want to estimate the distribution of the salary of users of *LinkedOut*, a professional social network. Your task is to compare it to the data released by the Federal Statistical Office, which gives you a probability density function $f_X(x)$ of the salary X that looks like this:



1. **CCDF (4 pts)** First, you want to model the distribution shown above by a Pareto distribution. Estimate the parameters β and γ , and plot the CCDF, i.e. the function $P(X \geq x)$.



2. **Estimate the mean and the variance (6 pts)** Now you want to compute an unbiased estimate of the mean and the variance of *LinkedOut* users' salary. Complete (using pseudo-code) the algorithm that computes *both* quantities (mean and variance) using a single random walk.

```
curr_node = start_node
wsum = 0.0 // sum of users' salary weighted by inverse degree
wss = 0.0  // sum of squared users' salary weighted by inverse degree
dsum = 0.0 // sum of inverse degrees

for i = 1, ..., N do:
    curr_node = random_neighbor(curr_node) // Select a random neighbor
    // complete the loop
```

```
endfor
```

```
mean_estimate = // <- complete
var_estimate =  // <- complete
```

3. **Convergence of the mean and the variance (4 pts)** It turns out that the distribution of the salary of *LinkedOut* users is the Pareto distribution given above. If your algorithm ran for a very long time on a very large graph, what would your mean and variance estimators converge to?