

Yury Lifshits

Steklov Institute of Mathematics at St.Petersburg

November 2006

1 / 33

To find and state key open algorithmic problems for future web technologies

2 / 33

Outline

- 1 Intro: Criteria and Questionnaire
- 2 Problem 1: Large-Scale Filtering
- 3 Problem 2: Large-Scale Matching
- 4 Problem 3: Tag Propagation
- 5 Problem 4: Structure Discovery

3 / 33

INTRO

What are **my personal** criteria for choosing open problems?

What kind of questions should I answer about proposed problems?

4 / 33

Criteria

- Ultimate relation to technology challenge
- Familiarity with the corresponding applied field
- Interplay of several basic fields
- Freshness (hence, badly formalized)

I do not use:

- Difficulty
- Popularity and age of the problem
- Famous author

Your favorite criteria?

5 / 33

Questionnaire

- Technology challenge?
- Sample formalization?
- Basic fields involved?
- Research workflow?
- Your constructive feedback?
- References? Similar Ideas? [To be done]

6 / 33

Disclaimer

My style is

- At first, think independently (e.g. pose new problems)
- **Only after that** look into literature

Hence, the following problems might be already known and heavily studied!

7 / 33

PROBLEM 1

**Large-Scale Filtering**

What are the fastest algorithms for personal news aggregation?

8 / 33

## 1.1. Challenge

Personal news aggregation:

Every user has a preference profile:  
specified information sources, keywords, tags(topics),  
popularity, references to the preferences of others

Every news item has its own description:  
text, votes and recommendations, tags,  
author reputation, comments

**Filtering problem:**

To find, say, ten most appropriate news items  
for every user

9 / 33

## 1.2. Formalization

- Every profile is a normalized **red** vector (point on sphere) in  $n$ -dimensional space
- As well, every news description is a normalized **blue** vector in the same space
- We use cosine measure (scalar product) for similarity
- Computational problem: after preprocessing all **blue** points, for every incoming **red** point compute quickly ten closest **blue** points

Data structures for storing all profiles and all news?

10 / 33

## 1.3. Fields Involved

- Text classification, kNN algorithms
- Computational Geometry
- Data Structures
- Compression (sparse sets)
- Linear Algebra (singular decomposition trick)
- What else?

11 / 33

## 1.4. Workflow

- Find fast algorithms for all-to-all filtering problem
- Suggest data structures for storing profiles and news
- Study filtering in dynamic settings: with profiles and descriptions quickly evolving in time
- Describe spam prevention mechanisms for large filtering systems

12 / 33

## 1.5. Constructive Feedback

Do you know related results?

What is the most important theoretical question in this problem?

How to make my formalization better?

13 / 33

## PROBLEM 2

### Large-Scale Matching

What is the most effective algorithm for distributing sponsored links among all websites?

14 / 33

## 2.1. Challenge

**Effective sponsored links (ads) distribution:**

Every ad has a target description  
Every website has an audience description

**Business objective:**

Maximize ratio clicks/displays

15 / 33

## 2.2. Formalization

- Every website's audience profile is a normalized **red** vector in  $n$ -dimensional space
- As well, every ad target is a normalized **blue** vector in the same space
- We use cosine measure for similarity
- Computational problem: compute **matching** between ads and websites that satisfy some constraints and **minimize the sum of distances** (ad - website)

16 / 33

## 2.3. Fields Involved

- Computational Geometry
- Linear Algebra (singular decomposition trick)
- Data Structures
- Compression (sparse sets)
- Game theory
- Optimization
- What else?

17 / 33

## 2.4. Workflow

- State ads distribution as an optimization problem
- Find algorithms that can approximately solve this problem faster than  $(\#websites) \times (\#ads)$
- Introduce feedback to the model: after every click on any ad we receive some additional knowledge about the world and can use it for improvement of our matching

18 / 33

## 2.5. Constructive Feedback

Do you know related results?

What is the most important theoretical question in this problem?

How to make my formalization better?

19 / 33

## PROBLEM 3

### Tag Propagation

How to extend partial categorization of websites to the whole web?

20 / 33

## 3.1. Challenge

### Web categorization:

People use millions of keywords (tags)  
There are billions of webpages  
We have **very sparse** training collection of pairs (website, tag)

### Goal:

Get a fast algorithm that can characterize any given website

### Applications:

Ads targeting  
Search results annotations  
Automatic web directories

21 / 33

## 3.2. Formalization

- We have the graph of hyperlinks
- Fix a tag. For every initially labelled website let  $T_0(i) = 1$ , for others  $T_0(i) = 0$
- Then we use recursive equation and take a limit:

$$T_k(i) = T_{k-1}(i) + \alpha \sum_{j \text{ links to } i} T_{k-1}(j)$$

- Computational problem: use some preprocessing for initial tag distribution and then for every given website compute quickly ten tags with the highest rank

22 / 33

## 3.3. Fields Involved

- Data Structures
- Compression (sparse sets)
- Numerical Analysis (speed of convergence)
- What else?

23 / 33

## 3.4. Workflow

- Define formulas for tag "propagation"
- Construct a fast algorithm for computing, say, ten most relevant tags of arbitrary website

24 / 33

### 3.5. Constructive Feedback

Do you know related results?

What is the most important theoretical question in this problem?

How to make my formalization better?

25 / 33

### PROBLEM 4

#### Structure Discovery

Consider keywords we use in everyday life. Can we suggest an algorithm that computes the most appropriate hierarchy of these keywords?

26 / 33

### 4.1. Challenge

We can collect many huge data sets:  
call graphs, shopping histories, search histories  
social networks, RSS subscription graph  
HOW TO BENEFIT FROM THEM?

Example: **hierarchy discovery**

We have some **folksonomy**

How to compute "optimal" tags hierarchy?

#### Applications:

Visualization and better navigation

Solving synonymy problem

27 / 33

### 4.2. Formalization

- Every tag is characterized by corresponding set of websites
- We want to compute the optimal **AND-OR** tree of tags
- Optimal means minimal correctness violation
- Correctness: sons of OR vertex should be disjoint, parent set contains children sets, etc...

28 / 33

### 4.3. Fields Involved

- Computational Biology (phylogeny algorithms)
- Approximate algorithms
- What else?

29 / 33

### 4.4. Workflow

- Fix a format of tag description and define an optimality criteria for hierarchy of tags
- Construct a fast algorithm for computing optimal hierarchy
- Study interplay with algorithms for constructing phylogeny tree

30 / 33

### 4.5. Constructive Feedback

Do you know related results?

What is the most important theoretical question in this problem?

How to make my formalization better?

31 / 33

### Voting

We discuss four problems. Which one do you like the most?

- Large-Scale Filtering
- Large-Scale Matching
- Tag Propagation
- Structure Discovery

32 / 33

## Main points

My homepage: <http://logic.pdmi.ras.ru/~yura/>

### Today we learn:

- Technology challenges: personal aggregation, effective ads, usage of huge data collection
- Key algorithmic challenge: large-scale algorithms that are faster than naive (usually quadratic) approaches
- Next steps: (1) survey, (2) formalizations, (3) public discussion

Thanks! Questions?