GOSSPLE: personalized and decentralized queries

Anne-Marie Kermarrec
INRIA, Rennes

With X. Bai, M. Bertier, A. Boutet, D. Frey, K. Huguenin, V. Leroy, A. Moin, G. Tan (INRIA) & R. Guerraoui (EPFL)

The Web revolution

Web content is generated by you, me, your friends and millions of others

(Two faces of) social networking has taken off at an unexpected scale and speed

Interest-based Web 2.0

- Users freely express their interest
- Personalized selection (LastFM)
- Collaborative tagging systems (Folksonomies) e.g. Delicious, Flickr, CiteUlike

There is a gold mine of information out there

Niche content relevant to small communities

How to find an answer to any ultra specific query?

A real-world example

English-speaking Rennes

Accomodation

Baby-sitting

Alice's family

Baby-sitting company, not even located in Rennes

Same request from a family located in Seine et Marne

General ad site

Google

"Baby-sitter anglophone Rennes"

1- Offre de job étudiant : Baby sitter anglophone
2- Archive11 Baby sitting Nourrice 87 000 ANNONCES GRATUITES de Baby ..
3-Recherche/ offres de service - Les dernières annonces

Request in English of a French-speaking baby-sitter

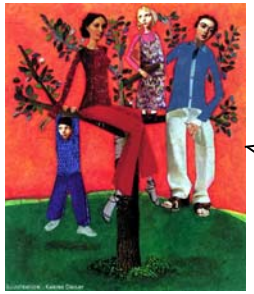
General ad site

« English baby-sitter Rennes »

Standard ad in French in Rennes

1- Annonce Niñeros(as) Rennes : baby-sitter jeune, dynamique et ...
2- Annonce Baby sitting Rennes : Baby-Sitter de 22 ans expérimentée ...
3Archive36 Baby sitting Nourrice 87 000 ANNONCES GRATUITES de Baby ...

State of the art solution

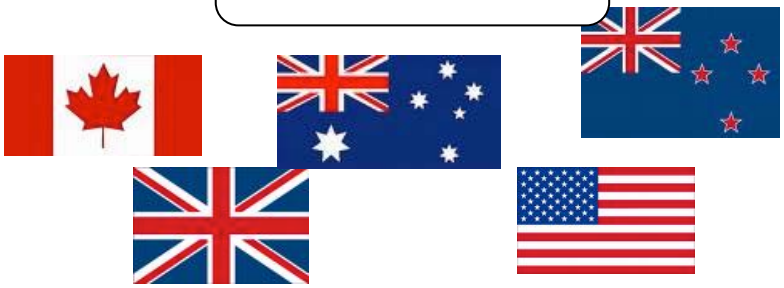


Mailing lists
Adequate forums

Seeking for a English-speaking baby-sitter against accommodation

Teaching assistants in Rennes

English-speaking students



Where to post the request ?

Need a Personal Perspective

- I can ask Facebook friends
- I can shout my need on Twitter

- What if none of my friends has English-speaking kids?
- How to find the right information?

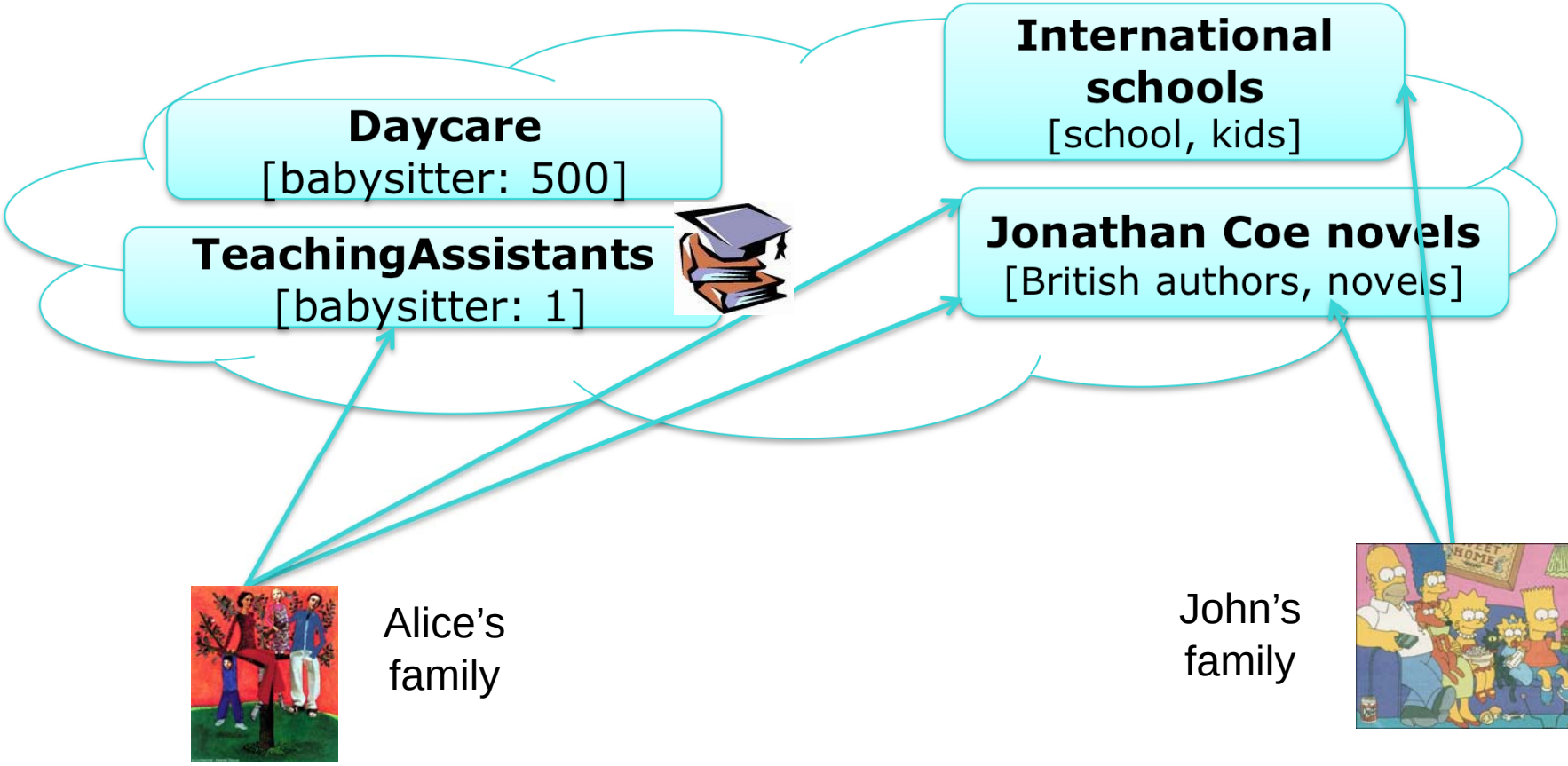
Leveraging implicit social links

Personalization through unknown acquaintances

The baby-sitter example

What would be ideal?

User-centric query



What if John and Alice have opposite taste in music?

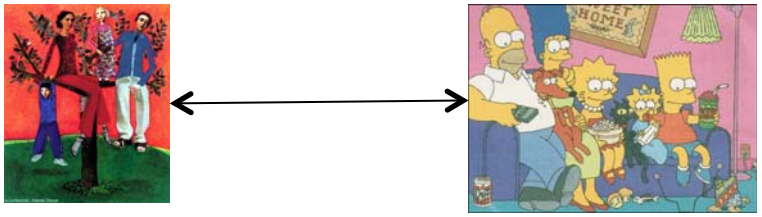
Need to cover **multiple interests** through multiple “friends”

Why is this difficult?

Why automatisation is difficult?

- Users are not willing to spend a lot of time specifying their needs
- Need to capture users preferences on some data

- Obvious data are **profiles** in interest-based Web 2.0 applications



Personalisation calls for decentralization

Anonymity: fighting the Big Brother's attitude

- e.g. New terms of uses of Facebook (2009), Beacon feature of Facebook (2007)

Scalability / Reactivity

- Enable to manage metadata at a user's granularity
- Cope with dynamics

Complex without global knowledge

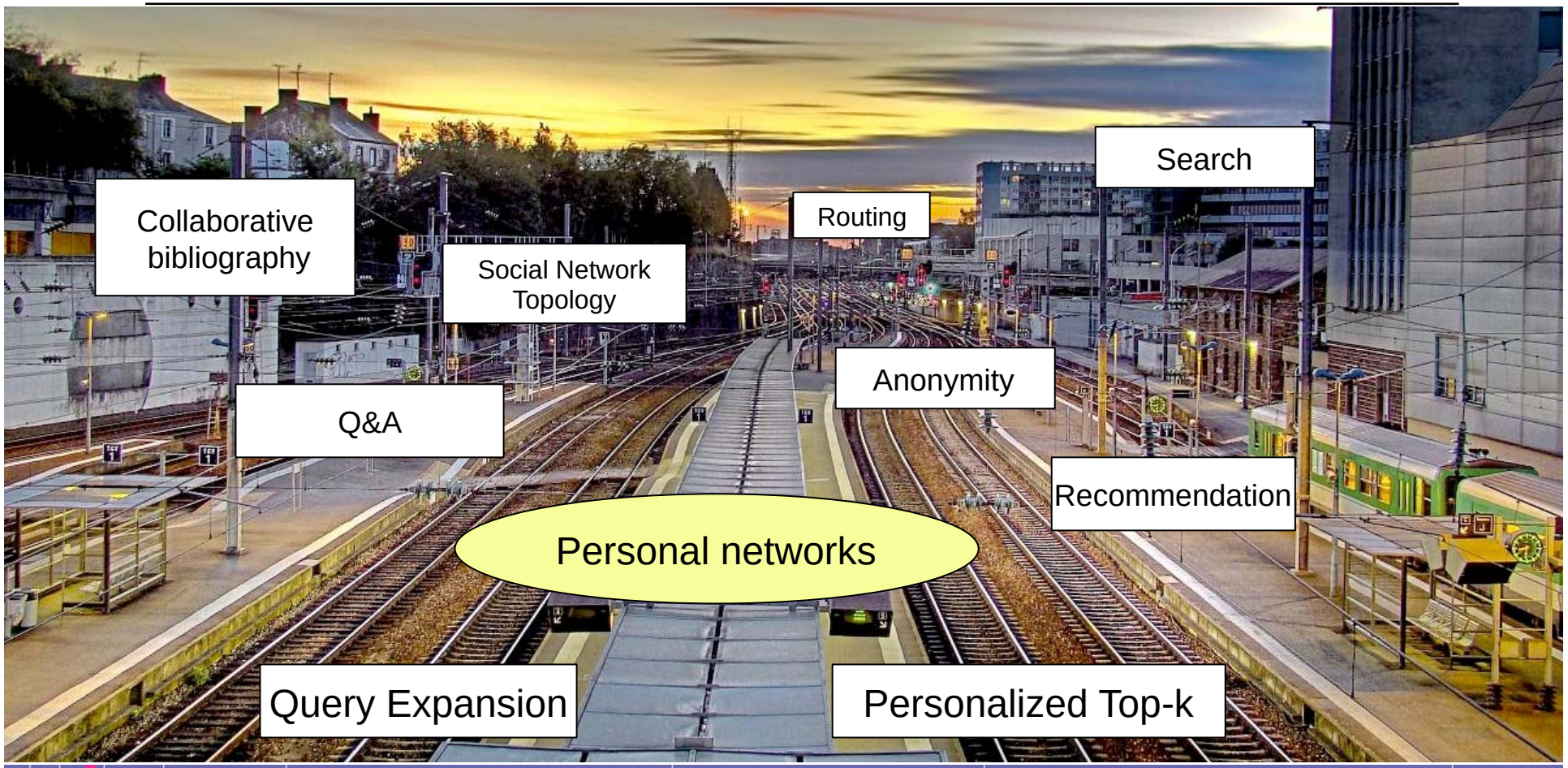
What Gossple is about?

GOSSPLE in a Nutshell

Personalized approach to navigating the digital world : Favor individuals as opposed to large masses

Decentralized approach to provide scalability, reactivity and privacy

Gossple's (Current) Tracks

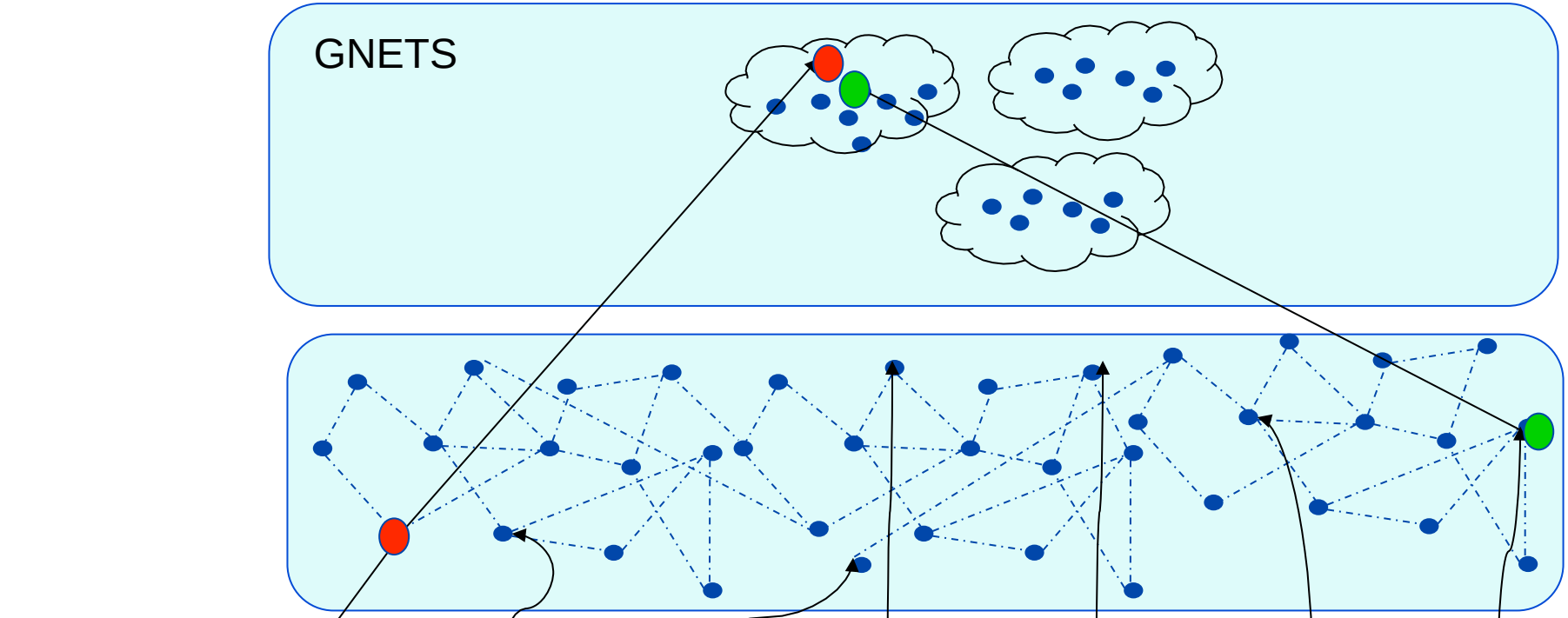


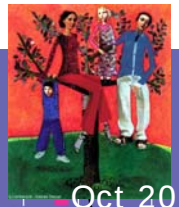
Gossple's Central Abstraction

The Gossple personal network: GNET

Leveraging wisdom of the **reduced but right** crowd

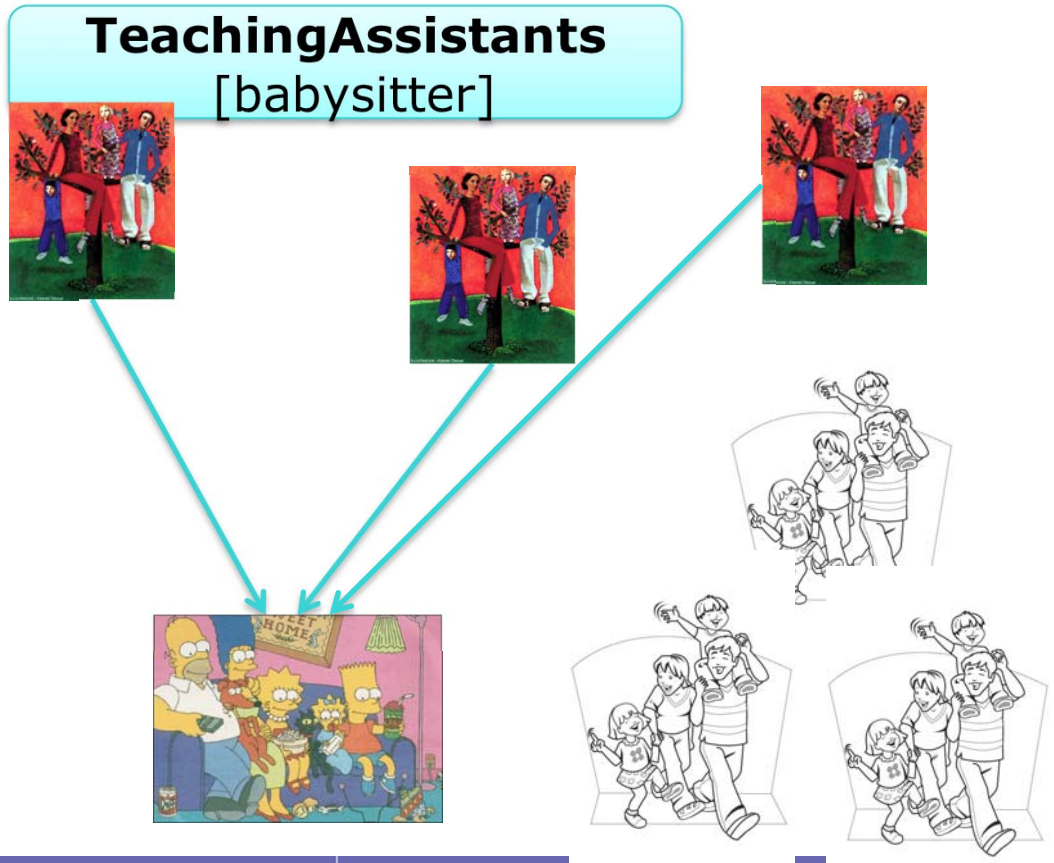
Building up the personal network





Application: query expansion

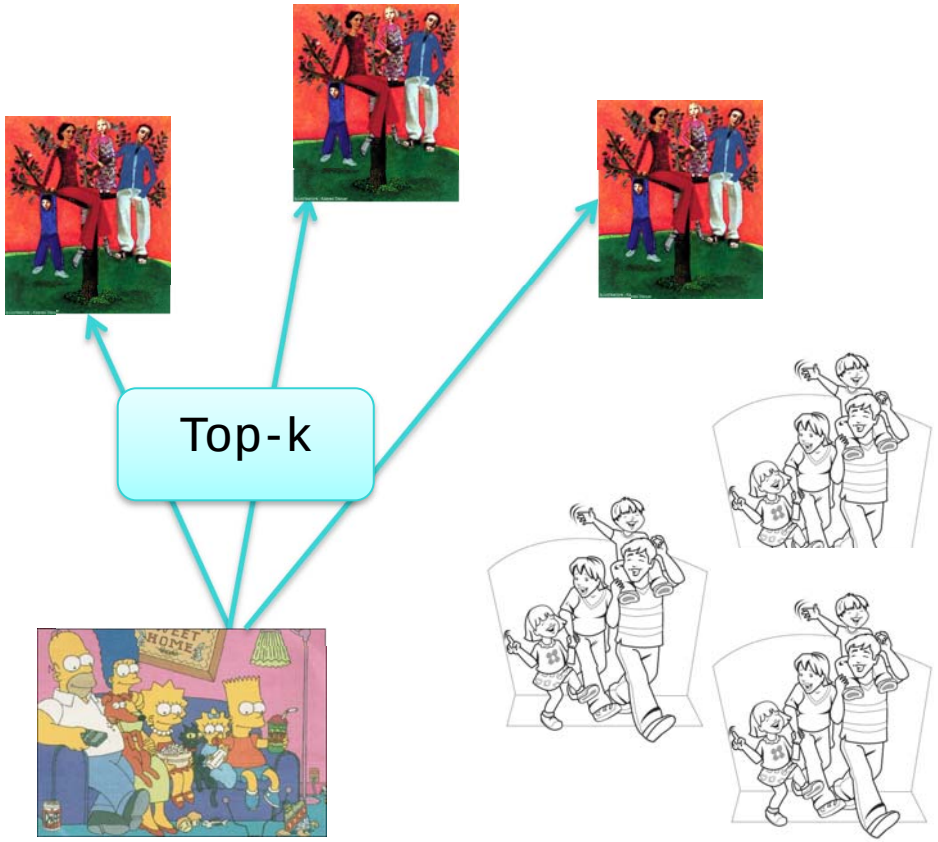
- Differentiate between apples or jaguars
- Find a baby sitter?



Application: Top-k

- Consider a subset of the network
- No need for pre-computed inverted lists

Partitioned query processing



Personal networks: contributions

- Capture affinities
  - Application-based metric
  - Multi-interest
- Discover relevant nodes
  - Gossip-based maintenance
  - Efficient maintenance: Bloom filters

Creating the GNET

Maintaining a view of the k “best” nodes according to a metric

- which ones are the best?
- how to discover them?

Which ones are the best?

Similar users

- Interested in the same items
- Using the same tags
- Stronger metrics (same tags on same items)

Overlap

- Weak metric (wrt size of the profile)
  - **u1** i1 , i2 , i3
  - **u2** i1 , i2
  - **u3** i1 , i2, i3 …, i500

ItemOverlap(u1 ,u2)=2 < ItemOverlap(u1 ,u3)=3

Item Cosine Similarity

Normalized overlap

- bigger overlap increases the score
- no shared interests decreases it
- directly takes into account the weight of items/tags

$$\cos(\vec{v}_1, \vec{v}_2) = \frac{\vec{v}_1 \vec{v}_2}{\|\vec{v}_1\| \|\vec{v}_2\|}$$

$$ItemCos(\vec{u}_1, \vec{u}_2) = \frac{|Items(\{\vec{u}_1\}) \bigcap |Items(\{\vec{u}_2\})|}{\sqrt{|Items(\{\vec{u}_1\})| . |Items(\{\vec{u}_2\})|}}$$

Item Cosine Similarity: Example

Items (u) = |distinct (Item(u, i1), ...)|

Boolean vector

u1 i1 , i2 , i3 ⇒ ItemVect (u1) = (1, 1, 1, 0...)

u2 i1 , i2 ⇒ ItemVect (u2) = (1, 1, 0, 0...)

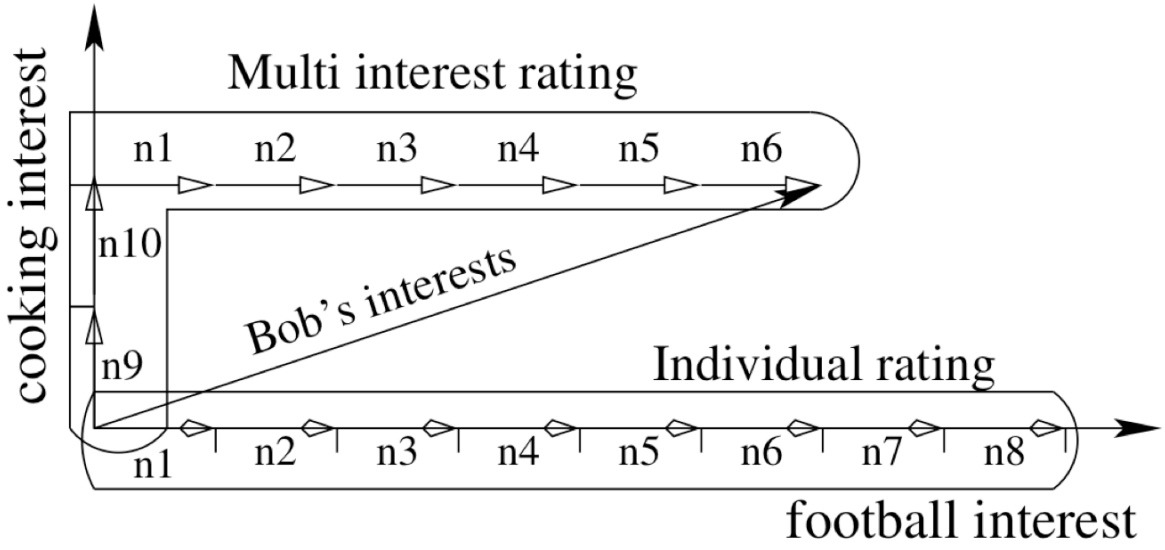
u3 i1 , i2 , i3 ..., i500 ⇒ ItemVect (u3) = (1, 1, 1, 1..., 1)

ItemCos (u1, u2) = 2 √≈ 0.8 >

ItemCos (u1, u3) = 3 √≈ 0.1

Coping with multi-interests

- Select a set of “most similar” users
- **Item cosine similarity:** favours specific and dominant interests



Multi-Interest Rating

- Rate the view **as a whole** instead of each potential neighbor
- Choose a set of neighbors that covers the user's interests

$$SetItemVect(set) = \sum_{p \in set} \frac{(ItemVect(p) \otimes ItemVect(n))}{\| ItemVect(p) \|}$$

Items of interest for nodes in Neigbhor(n)

Normalized not to take into account non shared interests

$$SetScore(n, set) = SetItemVect(set).ItemVect(n) * \cos(SetItemVect(set)., ItemVect(n))^{b}$$

Distribution

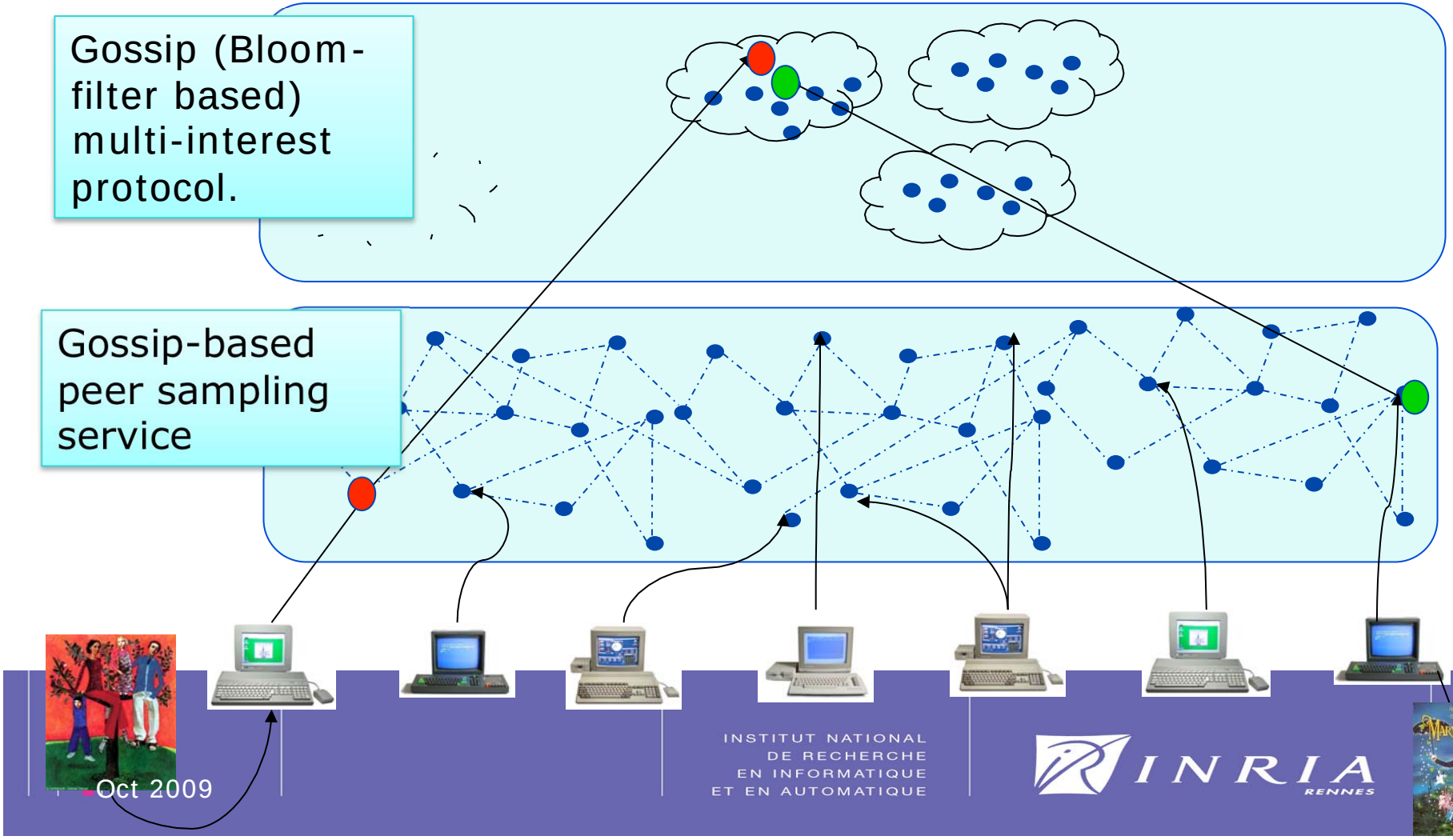
Building the personal networks (Gnets)

How to discover the best nodes?

How to cope with changes of interests or churn?

Gossip-based protocols

The networking infrastructure

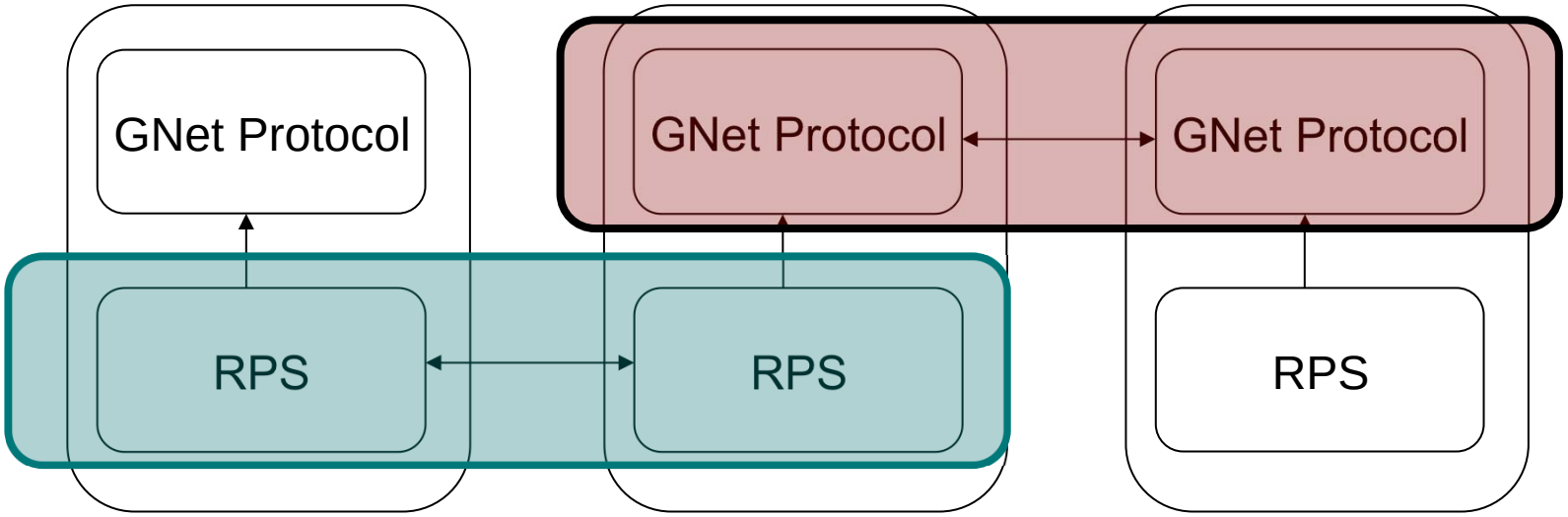


INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA RENNES

Gossiping framework

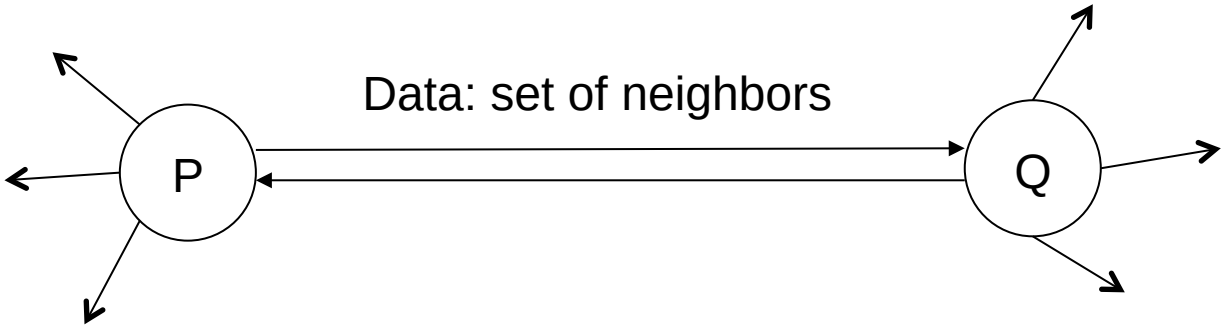
All nodes are examined: create a “small-world” like structure so that new nodes are discovered.



Gossip-based computing

Each peer maintains a set of neighbors

Parameter Space (Peer selection, Data exchanged, Data processing)



Data structures

GNet

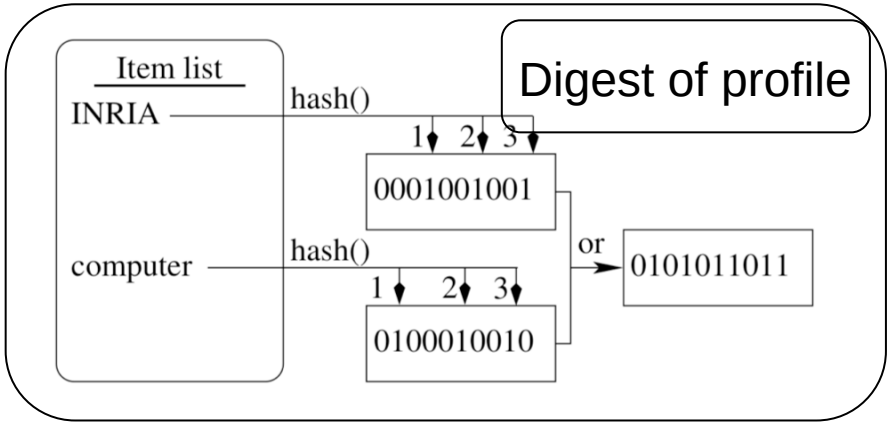
@IP:port	132.154.8.5:2020	. . .	@IP:port	113.121.15.12:920	. . .
Bloom Filter	010111011001		Bloom Filter	010110110101	
Profile	{(computer, X),(Paris, Y), (INRIA, Z), ...}				
Number of items	250		Number of items	124	
Update time	5		Update time	15	

c

RPS

@IP:port	102.14.18.1:2110	. . .
Bloom Filter	100100000110	
Number of items	300	
Update time	30	

r



Personal networks

- Personal network of n : GNet(*n*)

- When *n* encounters *q*
  - Evaluate distance between *n* and potential new view based on **set item cosine similarity** metric
  - Use of Bloom filters to limit the communication overhead

Multi-interest protocol

- Score of any combination: NP hard
- Heuristic: Starting from en empty view, builds the best view of size one, then two etc.

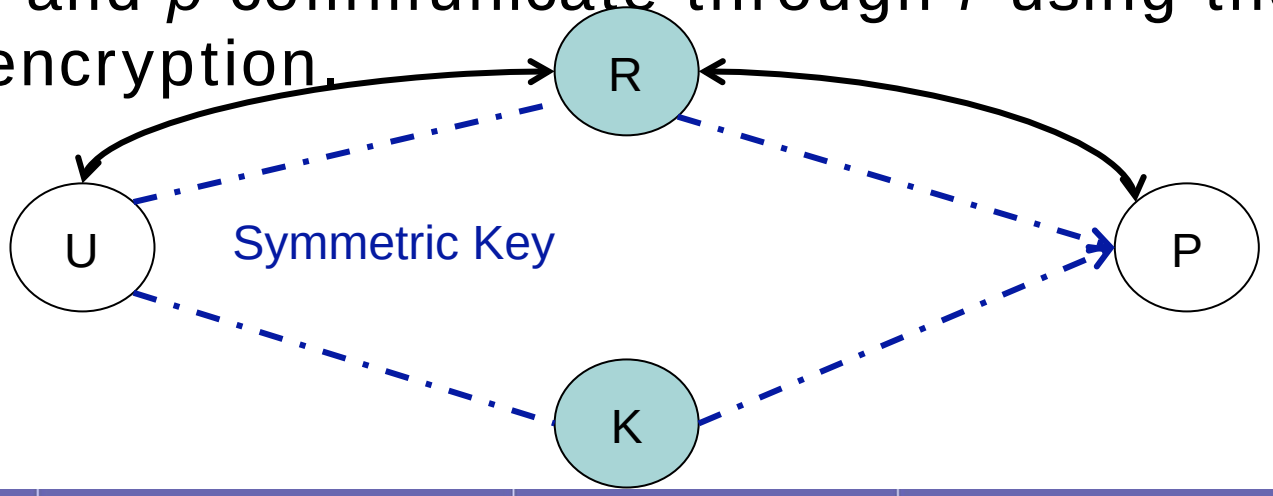
```
DataProcessing ()
    Bestview ={}
    For setSize  from 1 to viewSize do
        Foreach candidate in candidateSet do
            candidateView=bestview U {candidate}
            viewScore=SetScore(candidateView}
        bestCandidate = candidate that got the highest viewScore
        bestView= best View U {bestCandiate}
```

Gossip-based personal networks

- Clustering: use of Bloom filters
- Multi-interest clustering: heuristic
- Anonymity: no association between a user and her profile: **gossip on behalf**

Gossip on behalf

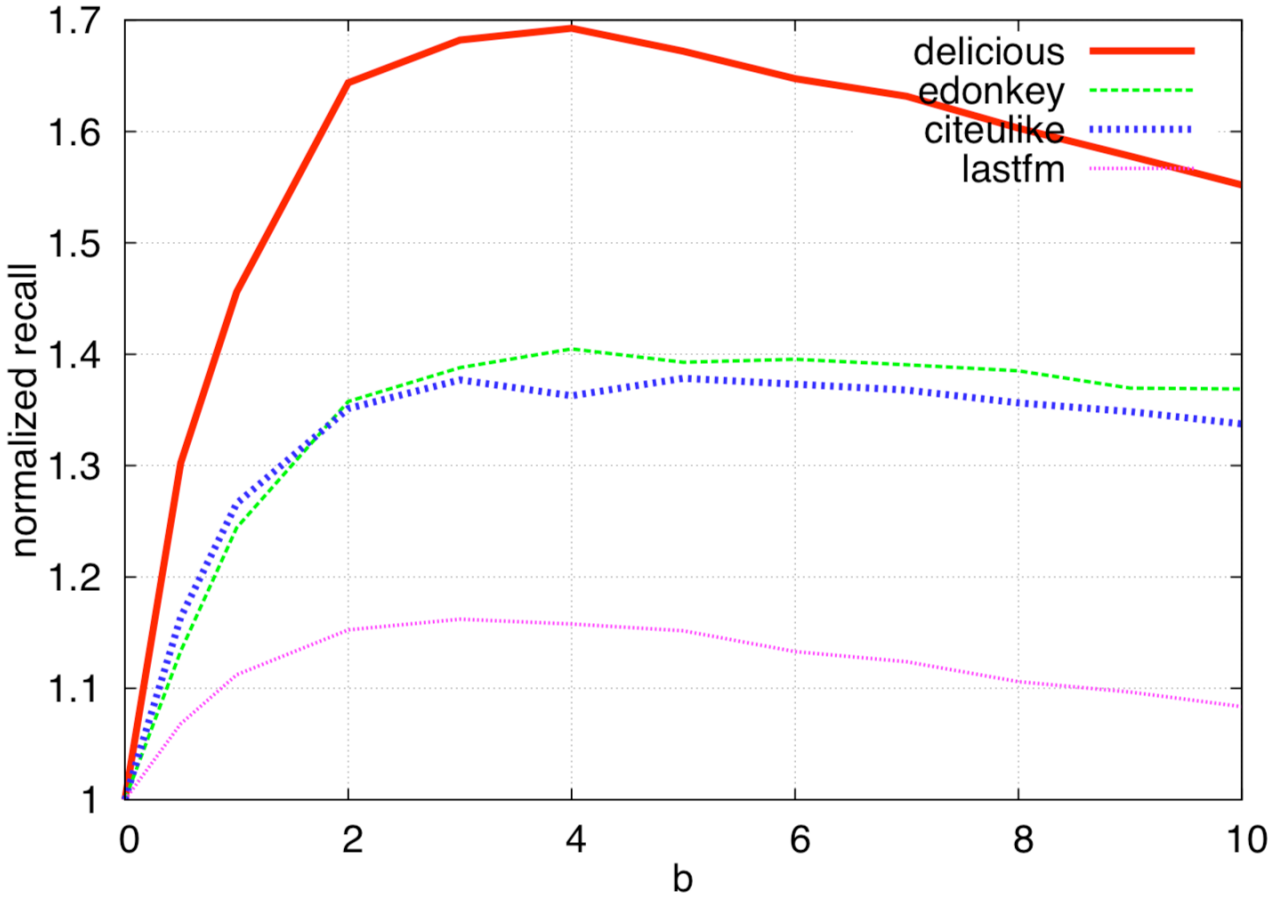
1. *n* chooses *p*, *r*, *k* at random (RPS)
2. *n* generates a symetric encryption *key*
3. *n* splits *key* in two parts and sends it to *p* via *r* and *k*
4. *n* and *p* communicate through *r* using the encryption.



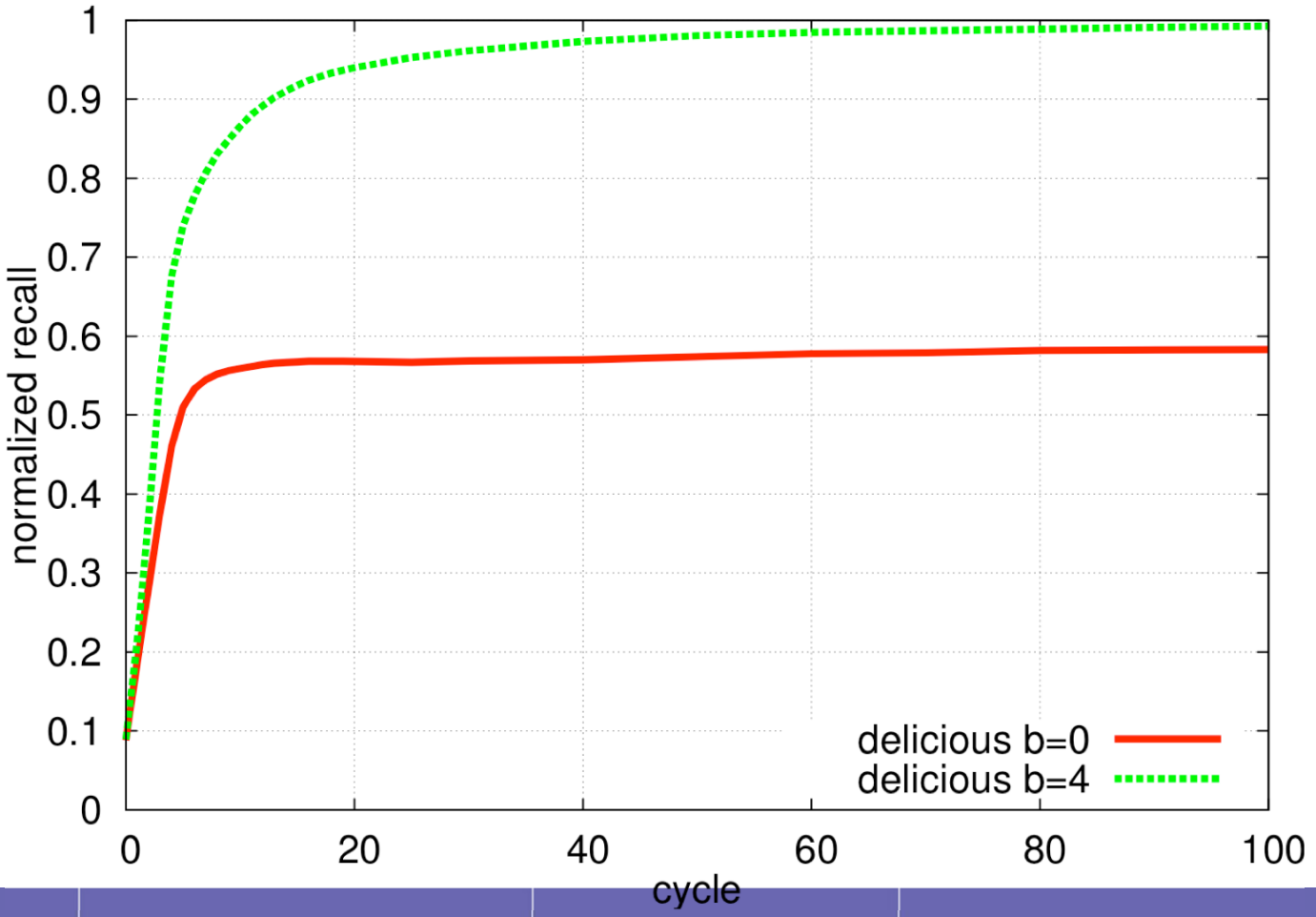
Evaluation

Traces from LastFM (1M), delicious (130,000), CiteUlike (33,000), Edonkey (137,000)

How good are Gossple friends?



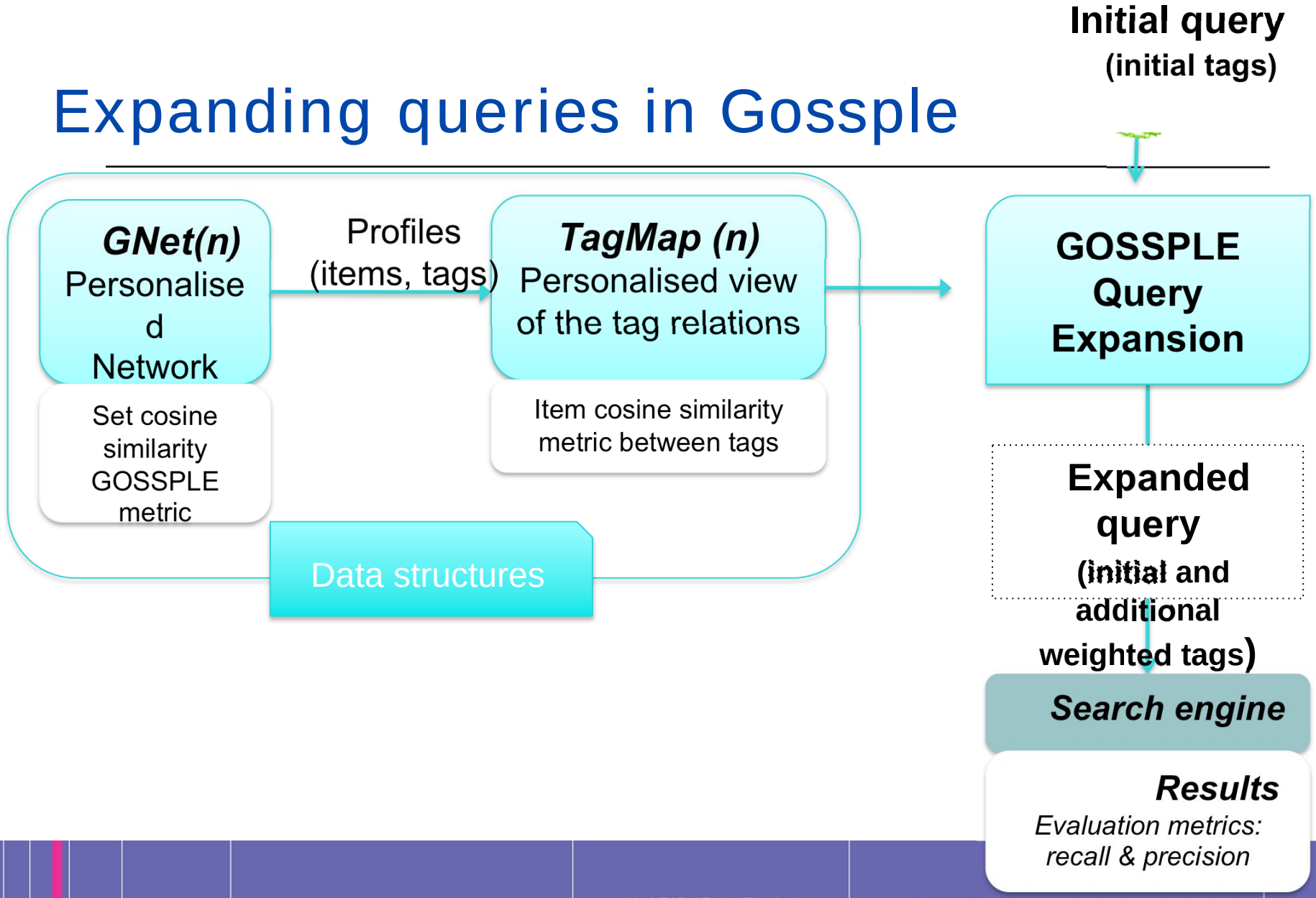
Cold start


normalized recall
cycle
delicious b=0
delicious b=4
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
20
40
60
80
100

Illustration

Query expansion

Expanding queries in Gossple



Personalized view of the world

The TagMap

The TagMap: tag/tag score

	Music	BritPop	Vivaldi	ColdPlay
Music	1	0.7	0.1	0
BritPop		1	0	0.7
Vivaldi			1	0
ColdPlay				1

Relating tags: Item tag cosine

Integration in the TagMap depending on the distance between user

- Neighbours(u): set of nodes in the personal network
- Item Cosine similarity: For each tag: number of occurences of the tag in Neighbours (u)

$$V_t\,[item_i] = v$$

$$TagMap[t_i, t_j] = \cos(\vec{V}_{t_i}, \vec{V}_{t_j})$$

Expanding Query

Direct reading (DR)

- Query expansion of size *q*
- Select the q tags scoring the highest

$$score_{DR}(t_i) = \sum_{t \in InitialQuery} TagMap[t, t_i]$$

- DR will never associate Music and Coldplay
- Worse DR could expand Music with Vivaldi

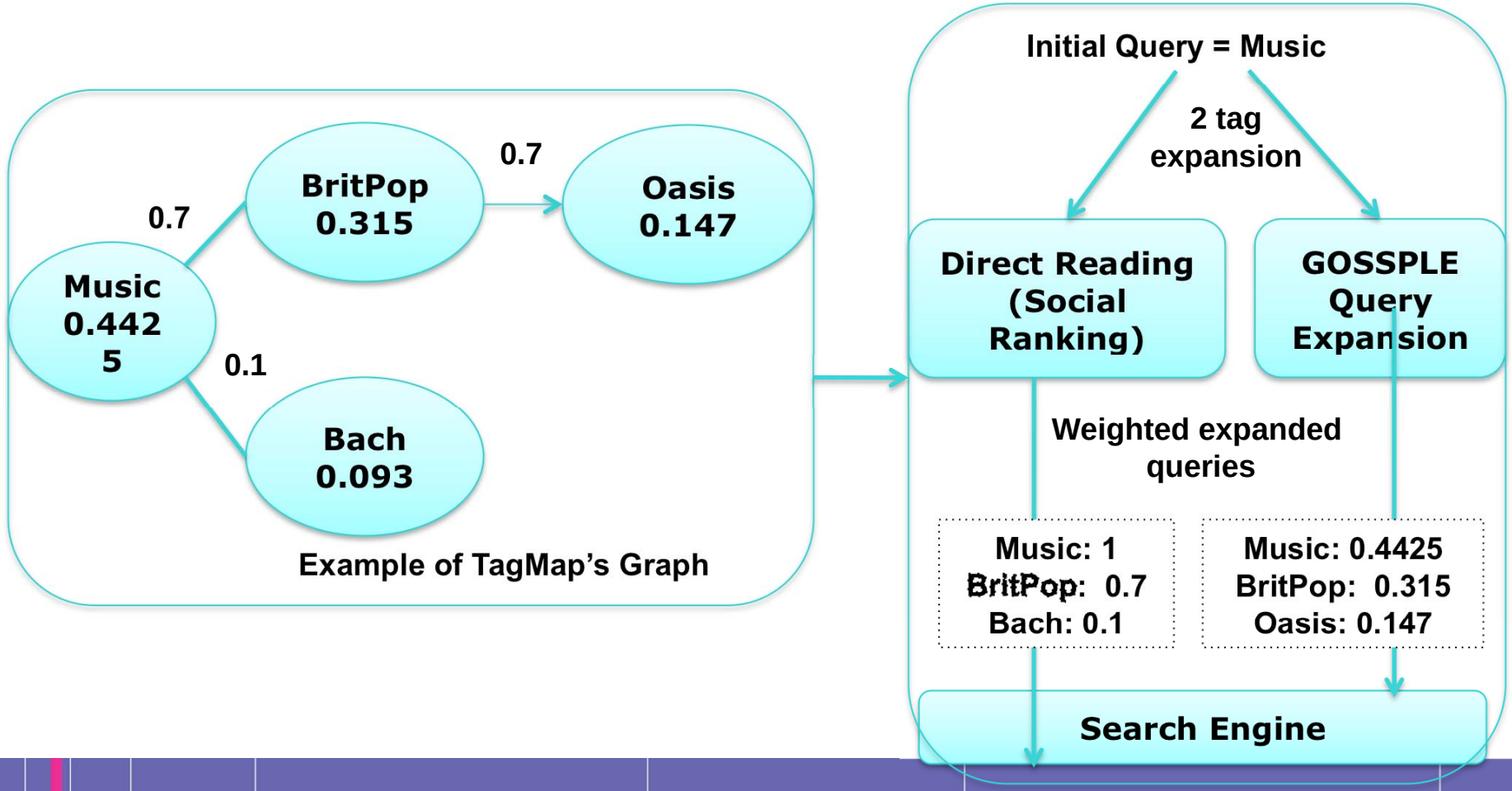
Expanding queries

Item sparsity: hidden relationships between tags

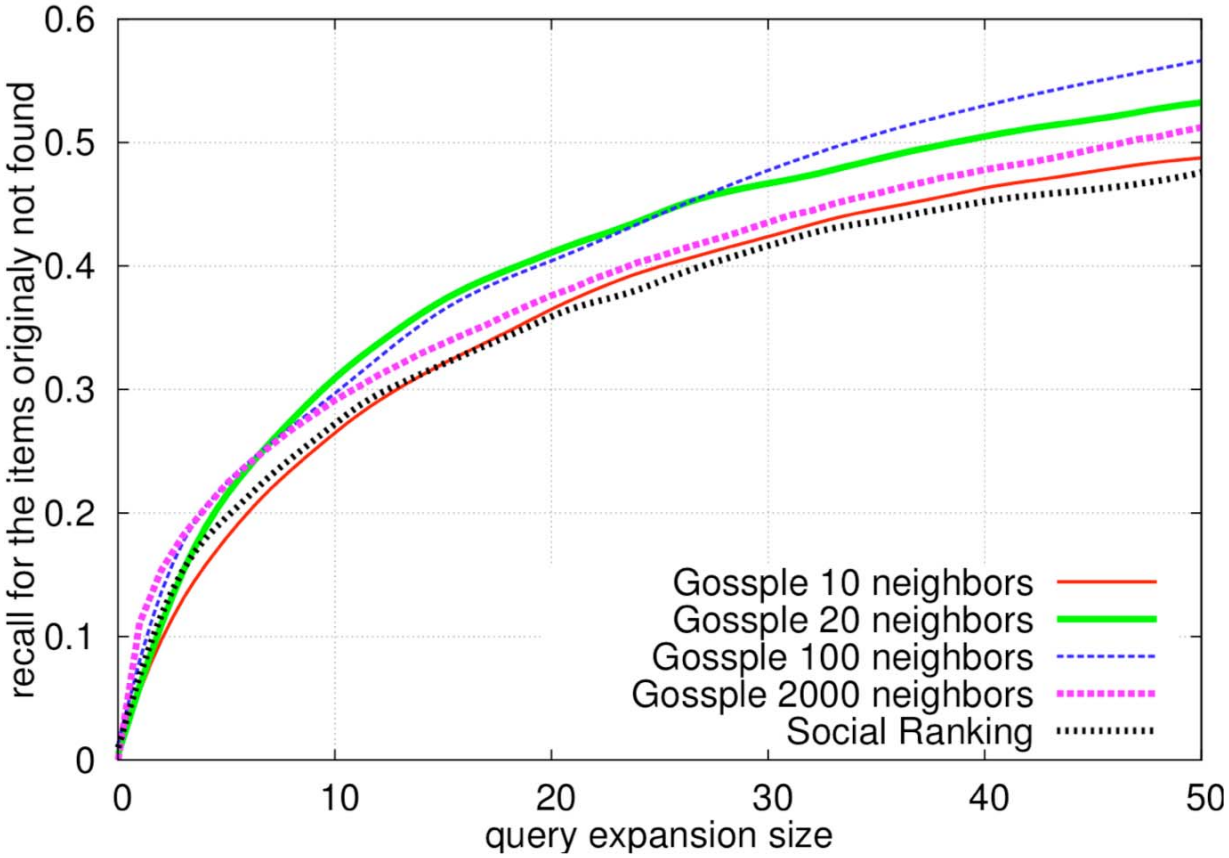
GRank: computation of tag *centrality*

- Adaptation of the PageRank algorithm
- Relative importance of a tag for a given tag and user

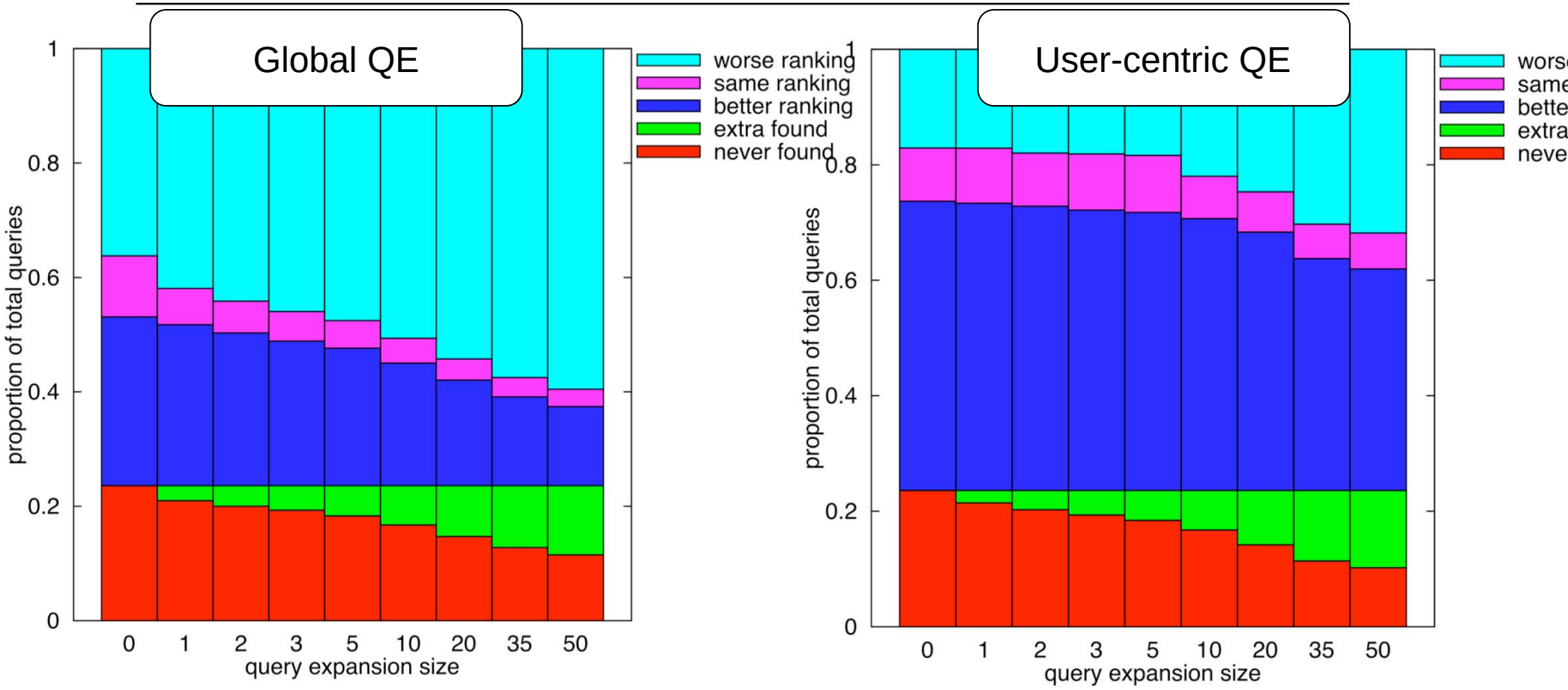
Expanding queries



Recall


recall for the items originaly not found
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10
20
30
40
50
query expansion size
Gossple 10 neighbors
Gossple 20 neighbors
Gossple 100 neighbors
Gossple 2000 neighbors
Social Ranking

Precision (delicious)



To take away

A case for user-centric approaches:

- **undeclared social connections (multi-interest)**
- **efficient and anonymous gossip protocol**

Applications

- **Query expansion**: harvest the personalized information, compute locally
- **Top-k processing**: discover the right helpers, compute remotely
- Recommendation/search

Thank you