LBSNRank: Personalized PageRank on Location-based Social Networks

Zhaoyan Jin, Dianxi Shi, Huining Yan, Quanyuan Wu, Hua Fan

National University of Defense Technology

Outline

- Scenario
- LBSNRank
  - Background;
  - Problem definition;
  - LBSNRank algorithm.
- Dataset
- Experimental Evaluation
- Conclusion and Future Work

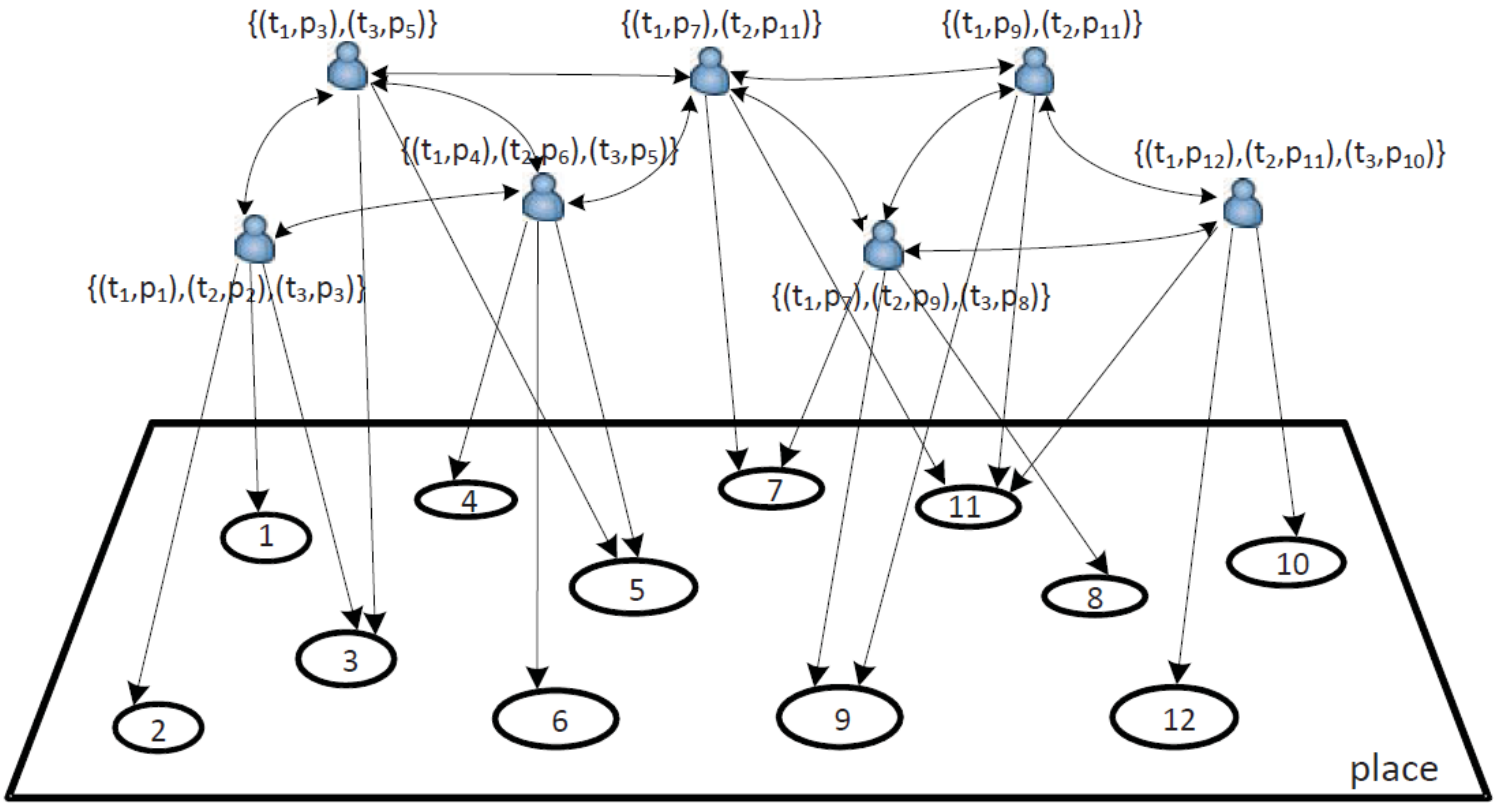
Scenario

As a traveler, you are prepared to visit the Great Wall in Beijing:

- In advance, you want to find some influential people in that area;
- After that, you may also visit some other popular places nearby.

Background

- A simple LBSN;



The LBSNRank problem

As time goes on, the location histories of users change, and thus the rankings of locations and users change, too. So

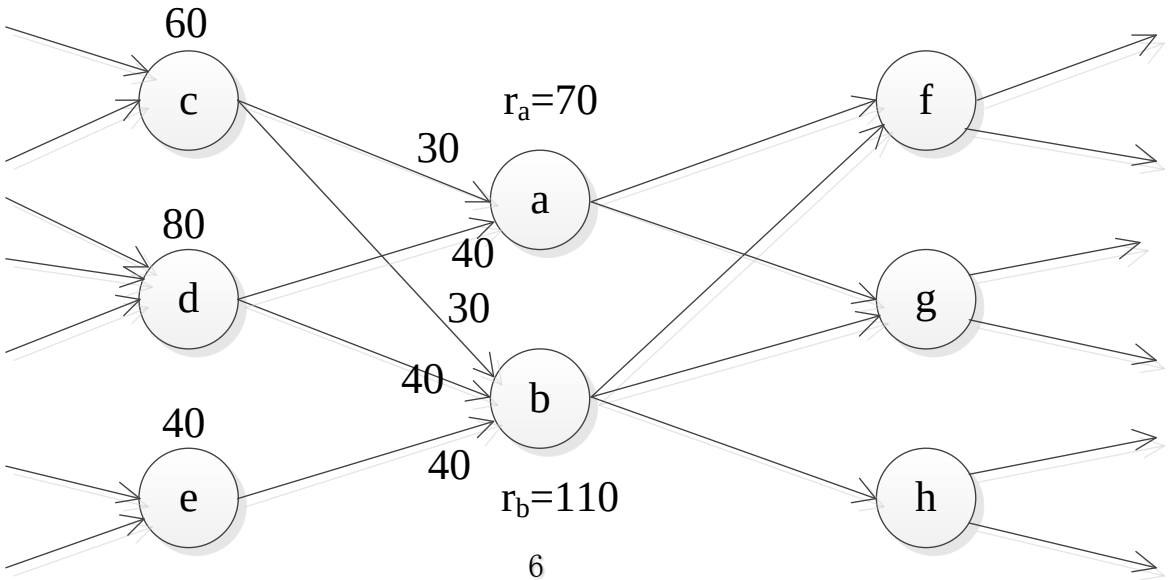
- How to decide the popularity of a location?

- And how to decide the influence of a user in a specific location?

The LBSNRank problem

Ranking of a user:

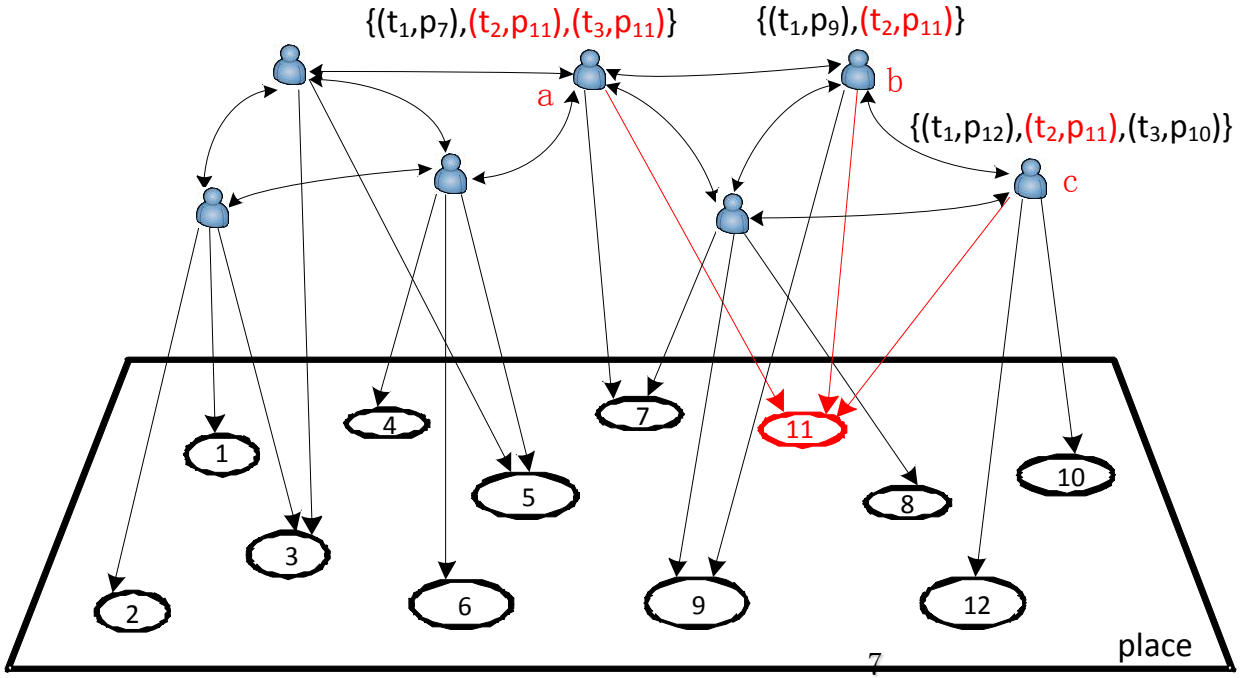
Definition 1. Given a social graph G and its corresponding location history L_G, the ranking score of a node u, in location p between t_1 and t_2 with the place-timestamp pairs $l_{u,p}(t_1, t_2) = \{(t_i, p)|(t_1 \leq t_i \leq t_2)\}$, is decided by its personalized PageRank: ($r_b > r_a$)



The LBSNRank problem

Ranking of a location:

Definition 2. The ranking of a location p in the period between t_1 and t_2 is proportional to the number of visitors, the ranking and visited times of each visitor, that is:



$$r_{11}=2r_a+r_b+r_c$$

The LBSNRank algorithm

1. Determining the location set S_p that will be preprocessed;
2. Computing the ranking of the location p in S_p;
3. Computing the rankings of people for each location p in S_p, and going to step 1 for next iteration.

The LBSNRank algorithm

- Node content changing → node rank changing (place) → place rank changing → S_p changing.

S_p

	P1	rank	P2	rank	P3	rank	P4	rank	…		Pm	rank
V_1	2	•	0	•	5	•	3	•	…		0	•
V_2	0	•	3	•	0	•	8	•			7	•
V_3	3	•	4	•	8	•	0	•			6	•
V_4	0	•	0	•	5	•	3	•			7	•
…	…	•		•		•		•	…			•
V_n	3	•	2	•	0	•	2	•			0	•

Dataset

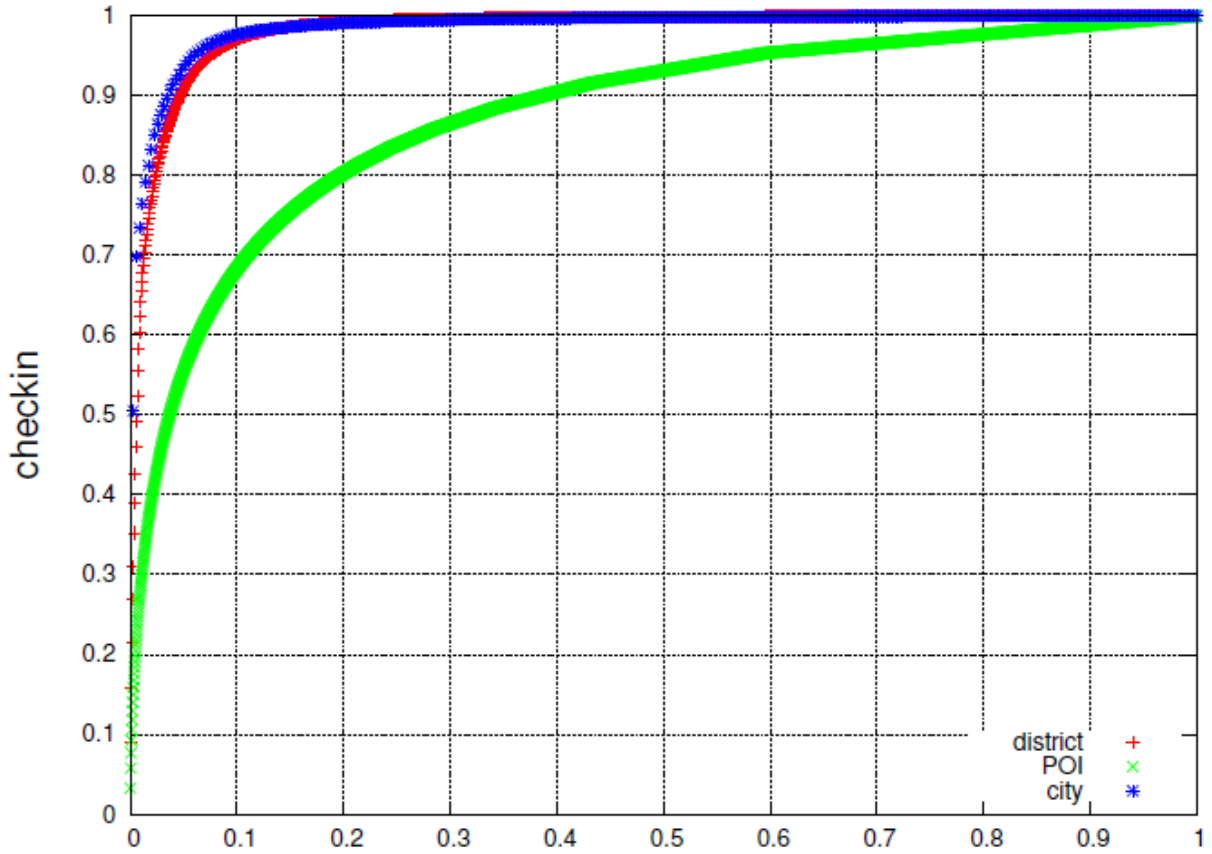
- To evaluate our method in a real LBSN, we have crawled the Dianping website to collect a dataset of users, their social links and checkin histories.

Table 1. Statistics of the Dianping Dataset

# users	# links	# checkins
204,074	926,720	2,730,072
# cities	# districts	# POIs
347	1,691	313,565

Dataset

- Location-Checkin Distribution



Experimental Evaluation

- Experimental Setup

  We implement the LBSNRank algorithm in Java on top of the Hadoop platform. Our experiments are executed on a cluster of 20 nodes, where each node is a commodity machine with a 2.16GHz Intel Core 2 Duo CPU and 1GB of RAM, running CentOS v6.0.

Experimental Evaluation

- Efficiency Evaluation

Figure 5. Comparison of Execution Time, where $\alpha = 0.1$ is the location ratio that we preprocessed offline

Experimental Evaluation

- Efficiency Evaluation

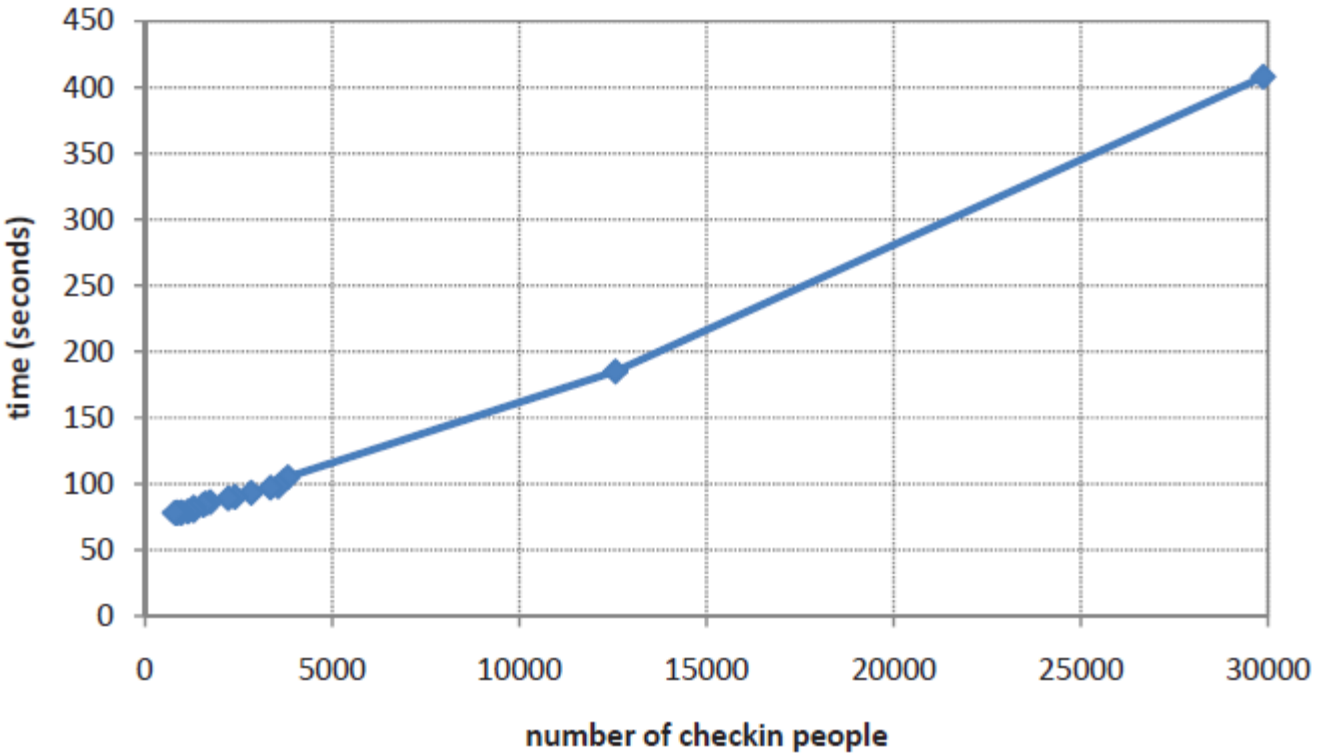
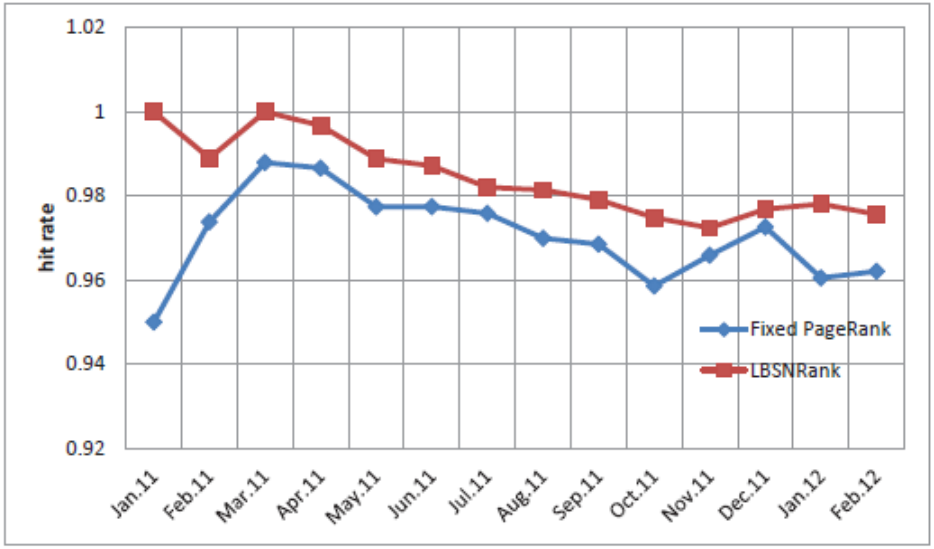


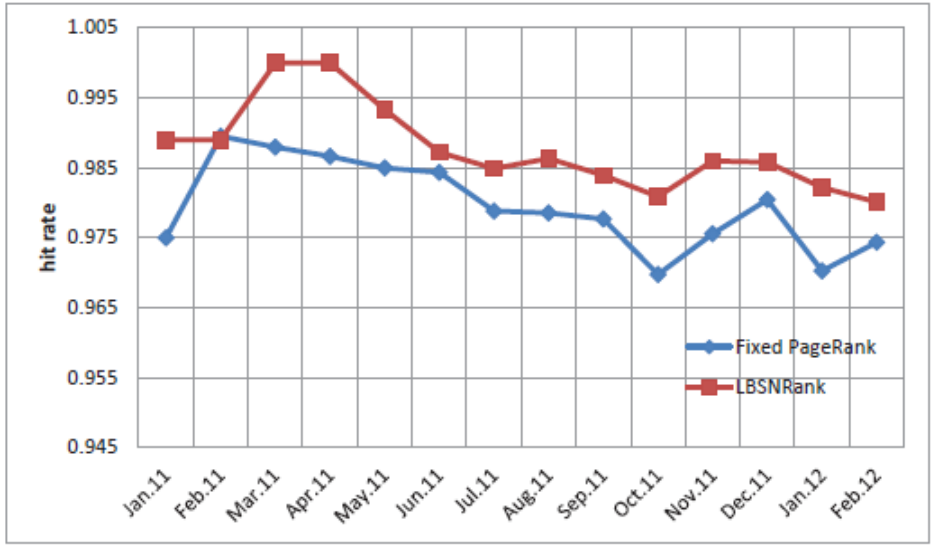
Figure 6. the Subgraph-Time Distribution

Experimental Evaluation

- Hit Rate Evaluation



(a) city, $\alpha = 0.1$



(b) district, $\alpha = 0.1$

Figure 7. Hit Rate Comparison, where $\alpha = 0.1$ is the location ratio that we preprocessed offline

Conclusion and Future Work

- Conclusion:
  - We study the problem of LBSNRank;
  - We propose the LBSNRank algorithm for rapid updates of users' locations;
  - We have crawled a dataset from the Dianping website;
  - We evaluate our experiments on Hadoop.
- Future Work:
  - How to answer users' queries timely with up-to-date results.

Thank you!

More information:

Zhaoyan Jin

jinzhaoyan@163.com