To Randomize or Not To Randomize: Space Optimal Summaries for Hyperlink Analysis

Tamás Sarlós,
Eötvös University and
Computer and Automation Institute,
Hungarian Academy of Sciences

Joint work with András A. Benczúr, Károly Csalogány, Dániel Fogaras, and Balázs Rácz

Contents

Personalized PageRank – Definition and Motivation

Definition: random surfer with *teleportation distribution* r and *tel. probab.* $c \approx 0.15$

$$\mathrm{PPR}_r(u) = c \cdot r(u) + (1-c) \sum_{v:(vu)\in E} \mathrm{PPR}_r(v)$$

Motivation: Search engines

- Improved ranking
- Fighting link spam

Slow to compute naively with the power method

Personalized PageRank – Linearity

Linearity:

$$\mathrm{PPR}_{\alpha_1 r_1+\alpha_2 r_2}(u) = \alpha_1 \mathrm{PPR}_{r_1}(u)+\alpha_2 \mathrm{PPR}_{r_2}(u)$$

Single page teleportation suffices:

$$\mathrm{PPR}_r(u) = \sum_v r(v) \cdot \mathrm{PPR}_v(u)$$

Personalized PageRank – Preliminaries

Two-phase algorithm

1. precomputes a PPR database
2. answers PageRank queries using the database

Exact PPR on a graph of $n \approx$ millions . . . billions of vertices:

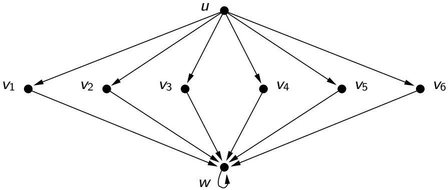
	Storage requirement	Person.
Topic sensitive [Haveliwala 02]	$O(t \cdot n)$ words	$t \approx 10 - 100$ topics
Hub decomp. [Jeh–Widom 03]	$O(h \cdot n)$ words	$h \approx 100.000$ pages
Lower bound of [Fogaras–Rácz 04]	$\Omega(n^2)$ bits, infeasible	all pages

Sampling Fully Personalized PageRank

- Express $\text{PPR}_u(v)$ as probability of random walk starting at u ending in v
- Sample ending points of random walks as above
- First algorithm with no restriction on u
- Additive error $\pm\epsilon$; out of bounds prob. δ
- Uses $O(n \cdot \epsilon^{-2} \log 1/\delta \log n)$ bits of space

Power Iteration and Dynamic Programming

Example



Power iteration amplifies the error downwards

Dynamic programming [Jeh–Widom WWW 2003] averages the error upward

$$\mathrm{PPR}_u^{(k+1)} = c\chi_u + (1-c) \cdot \sum_{v:(uv)\in E} \mathrm{PPR}_v^{(k)} / d^+(u)$$

Problem: small world, number of non-zeroes grow quickly in u's neighborhood

Rounded Dynamic Programming

Repeat $k_{\max} = 2 \log_{1-c} \epsilon$ times for all u

$$\widehat{\mathrm{PPR}}_u = \mathrm{Round}_k\Big(c\chi_u + (1-c) \cdot \sum_{v:(uv)\in E} \widehat{\mathrm{PPR}}_v / d^+(u)\Big)$$

- Space: n sparse PPR_u vectors in $O(n \cdot 1/\epsilon \log n)$ bits – optimal for top queries
- Can gradually decrease rounding error ϵ_k from $\epsilon_1 = 1$ to $\epsilon_{k_{\max}} = \epsilon$
- Deterministic output; inductive proof shows $\mathrm{PPR}_u(v) - 2\epsilon/c \leq \widehat{\mathrm{PPR}}_u(v) \leq \mathrm{PPR}_u(v)$
- Preprocessing: linear $O((n+m)/(c\epsilon))$ time

Dynamic Programming with Sketches

Drunken Surfer

- Mix up memories by random hash $h(v)$ of pages v

$$\mathrm{SPPR}_u(i) = \sum_{v:h(v)=i} \mathrm{PPR}_u(v) \quad \text{for } i = 1, \ldots, 2e/\epsilon$$

- Use surfers for $j = 1, \ldots, \log 1/\delta$ and use minimum vote: Count-Min Sketch [Cormode–Muthukrishnan 05]

$$\widehat{\mathrm{PPR}}_u(v) = \min_{j=1,\ldots,\log 1/\delta} \mathrm{SPPR}_u^{(j)}(h_j(v))$$

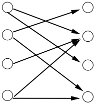
Dynamic Programming with Sketches Cont'd

- Dynamic programming over sketches by their linearity
- A variant also gives linear time preprocessing
- $O(n \cdot 1/\epsilon \log 1/\delta)$ bits of space – optimal for value queries

$$\mathrm{PPR}_u(v) - 2\epsilon/c - \epsilon \leq \widehat{\mathrm{PPR}}_u(v) \leq \mathrm{PPR}_u(v) + 2\epsilon/c$$

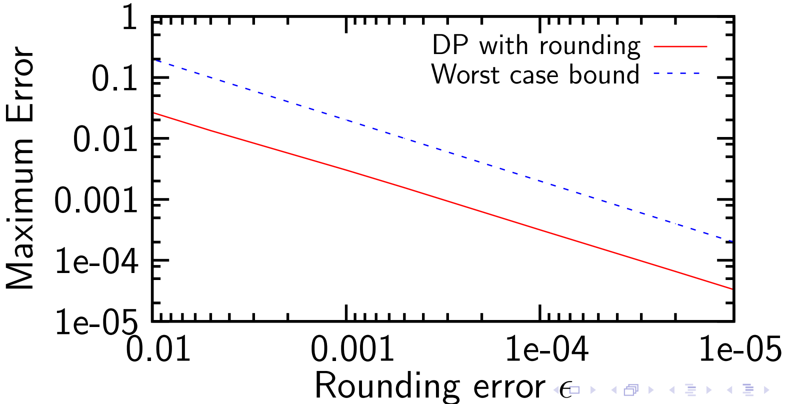
Lower Bounds

Reduction to one-way communication complexity of bit-vector probing

	Alice	Bob
	bit string $y \in \{0,1\}^{S}$	index i, output: y_i
1.	creates $G(y)$ 	
2.	transmits the PPR database of $G(y)$	
3.		queries the database for $\mathrm{PPR}_{u(i)}(v(i))$

Experiments

Stanford WebBase: $80M$ nodes, $800M$ edges
Measured accuracy over 1000 random nodes
Effect of rounding with $k_{\max} = 35$ iterations.



Quality of Approximate Rankings @ t

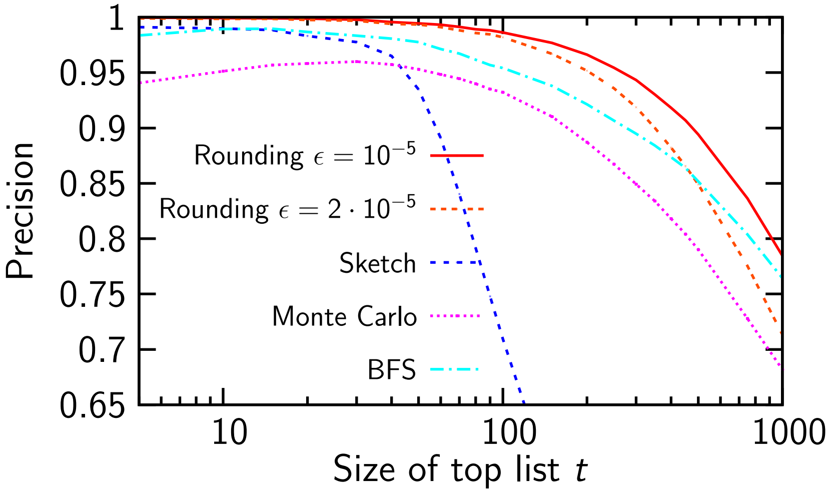
Precision = Recall:

$$\frac{|\text{approximate top-t} \cap \text{true top-t}|}{\text{t}}$$

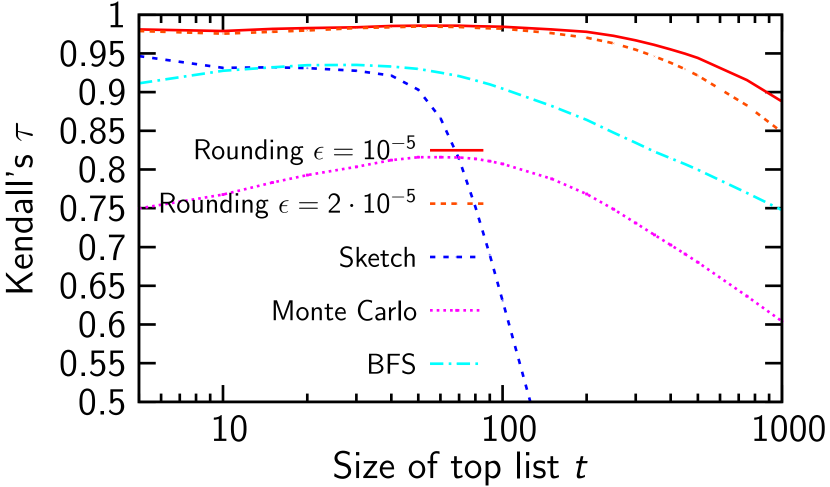
Kendall's Tau:

$$1 - 2\frac{\#\text{inversions in approximate top-t}}{\binom{\text{t}}{2}}$$

Precision


1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
Precision
10
100
1000
Size of top list t
Rounding $\epsilon = 10^{-5}$
Rounding $\epsilon = 2 \cdot 10^{-5}$
Sketch
Monte Carlo
BFS

Kendall's Tau



SimRank – Preliminaries and Sampling

"Two pages are similar if pointed to by similar pages" [Jeh–Widom 02]

$$\mathrm{Sim}^{(k)}(v_1, v_2) = \begin{cases} (1-c) \cdot \frac{\sum \mathrm{Sim}^{(k-1)}(u_1,u_2)}{d^-(v_1) \cdot d^-(v_2)} & \text{if } v_1 \neq v_2 \\ 1 & \text{if } v_1 = v_2. \end{cases}$$

$(1-c)^{k'}$-weighted path pair summation (incl. sampling [Fogaras–Rácz 05]) over

$$\begin{aligned} v_1 = w_0, w_1, \ldots, w_{k'-1}, w_{k'} &= u \\ v_2 = w'_0, w'_1, \ldots, w'_{k'-1}, w'_{k'} &= u \end{aligned}$$

SimRank – Reduction to Personalized PageRank

Version 0 reduction: count path pairs from v_1 and v_2 that may meet several times

$$\mathrm{Sim}^{(0)}_{v_1,v_2} = \sum_{k>0} (1-c)^k \sum_u \mathrm{RP}^{[k]}_{v_1}(u)\mathrm{RP}^{[k]}_{v_2}(u)$$

Recursively define self-similarity SimRank of *at least* $t+1$ inner meeting points as $\mathrm{SSim}^{(t+1)}(v)$

SimRank – Reduction to Personalized PageRank

Obtain SimRank by inclusion-exclusion of self-similarities

$$\mathrm{Sim}(v_1, v_2) = \sum_{k>0}(1-c)^k \sum_u \mathrm{RP}^{[k]}_{v_1}(u)\mathrm{RP}^{[k]}_{v_2}(u)\cdot\mathrm{SSim}(u)$$

$$\mathrm{SSim}(u) = 1 - \mathrm{SSim}^{(0)}(u) + \mathrm{SSim}^{(1)}(u) - \mathrm{SSim}^{(2)}(u) + \dots$$

Converges for $1 - c < 1/2$, technicalities to carry through approximation

Conclusion

- Efficient algorithms + lower bounds = space-optimal summaries for
  - Fully Personalized PageRank and for
  - SimRank with decay factor $< 1/2$
- At the heart of it: low space approximation of large vectors in the $\|\ldots\|_\infty$ norm
- Works well in practice

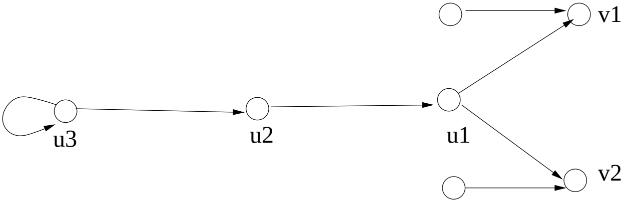
Thank you!

- http://www.ilab.sztaki.hu/websearch

Algorithms Compared

Algorithm	Running time
Dynamic Programming with $\epsilon = 2 \cdot 10^{-5}$ and $\epsilon = 10^{-5}$ rounding to varying ϵ_k	1.5 and 2.25 days
Dynamic Programming with $\epsilon = 6 \cdot 10^{-3}, \delta = 4 \cdot 10^{-3}$ sketches	6 days
Monte Carlo sampling with $N = 10000$ samples	6 days
Breadth First Search heuristic	3.5 days

SimRank Example



$$\sum_{k>0} \frac{1}{3^k} \sum_u \mathrm{RP}_{v_1}^{[k]}(u)\mathrm{RP}_{v_2}^{[k]}(u) = \frac{1}{4} \cdot \frac{1}{3} \left(1 + \frac{1}{3} + \frac{1}{3^2} + \dots\right) = \frac{1}{12} \cdot \frac{3}{2}$$

$$\mathrm{SSim}^{(0)}(u_i) = \frac{1}{3} + \frac{1}{3^2} + \dots = \frac{1}{2} \qquad \mathrm{SSim}^{(1)}(u_i) = \frac{1}{4}$$

$$\mathrm{SSim}(u_i) = 1 - \frac{1}{2} + \frac{1}{4} - \frac{1}{8} + \dots = \frac{2}{3} \checkmark$$