Model Reduction for Edge-Weighted Personalized PageRank

David Bindel

Mar 2, 2015

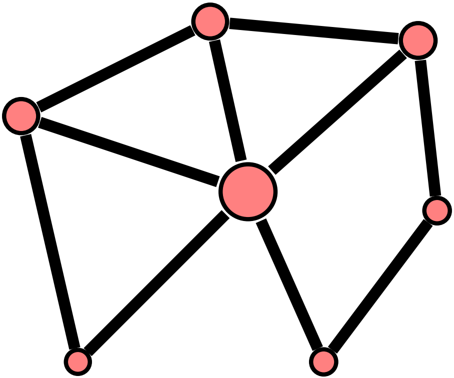
The PageRank Model

Surfer follows random link (probability α) or teleports to random node:

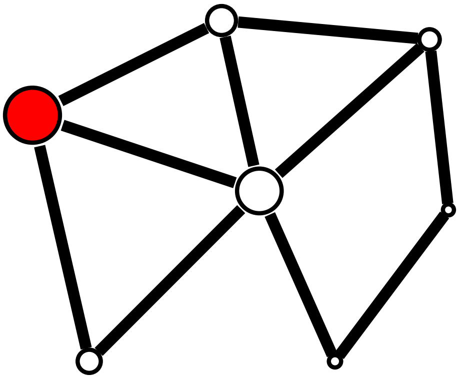
$$x^{(t+1)} = \alpha P x^{(t)} + (1 - \alpha) v,$$

$P = AD^{-1}$ is a (weighted) adjacency matrix with columns normalized.

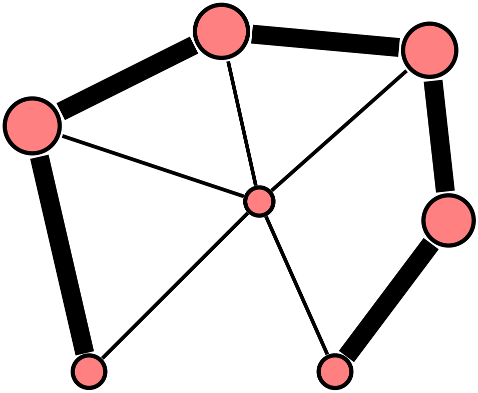
PageRank: Unweighted case



PageRank: Node Weighted



PageRank: Edge Weighted



The PageRank Model

Surfer follows random link (probability α) or teleports to random node:

$$x^{(t+1)} = \alpha P x^{(t)} + (1 - \alpha) v,$$

$P = AD^{-1}$ is a (weighted) adjacency matrix with columns normalized.

Stationary equations:

$$Mx = b, \quad M = I - \alpha P, \quad b = (1 - \alpha) v$$

PageRank iteration is a standard solver for this system.

Personalized PageRank

Introduce *personalization parameters* $w \in \mathbb{R}^d$, consider two cases:

$$\begin{aligned} &\text{Node-weight:} \quad && \mathrm{M}\ \mathrm{x}(\mathrm{w}) = \mathrm{b}(\mathrm{w}) \\ &\text{Edge-weight:} \quad && \mathrm{M}(\mathrm{w})\ \mathrm{x}(\mathrm{w}) = \mathrm{b} \end{aligned}$$

Examples:

- $b(w) = 1 - \alpha V w$, columns of V are authorities for reference topics
- Different edge types (authorship, citation, etc); w_i is weight of type i
- Nodes are writers, edge weights for topical similarity;
  w are weights in a weighted cosine similarity measure

Goal: Fast computation for varying w (different users, queries)

Edge-Weight vs Node-Weight

Node-weight personalization is well-studied

- Topic-sensitive PageRank: fast methods based on linearity
- Localized PageRank: fast methods based on sparsity

Little work on fast methods for edge-weight personalization!

Idea: Model Reduction

Replace large, expensive model by cheaper "reduced-order" model

- Common idea in physical simulations
- Use the model equations (vs black-box regression)
- Great for control, optimization, etc (many evaluations)
- Expensive pre-processing is OK

Model Reduction Framework

Observation: $x(w)$ approximately in a low-dimensional space:

$$x(w) \approx Uy(w), \quad U \in \mathbb{R}^{n \times k}, \quad k \ll n$$

Can find U by PCA/POD/KL/SVD on a "snapshot" matrix of samples

$$X = \begin{bmatrix} x(w_1) & x(w_2) & \dots & x(w_r) \end{bmatrix}$$

Can estimate quality of best approximation in the space

- A priori by interpolation theory (given bounds on derivatives)
- A posteriori from truncated singular values

Question: How to extract the best (or near-best) $y(w)$?

Background: Interpolation Connection

Why not go with interpolant

$$\hat{x}(w) = \sum_{j=1}^{r} x(w_j) c_j(w)$$

where $c_j(w)$ is some Lagrange basis for an interpolation space?

- Online phase is cheap
- Accuracy depends on Lagrange basis (Lebesgue constants)
- Have observed better accuracy with methods based on equations

Galerkin Approach

Goal:	$r = MUy - b \approx 0$ or $Uy \approx x$
Galerkin ansatz:	$W^T(MUy - b) = 0$
Bubnov-Galerkin:	$W = U$

Works great for *linear* parameterization

$$M(w) = I - \alpha P(w) = I - \alpha \left(\sum_i w_i P^{(i)} \right).$$

Model: pick edge type i with probability w_i, then pick edge of that type.

B-G system: $\tilde{M}(w)y(w) = U^T b$, where

$$\tilde{M}(w) = U^T M(w) U = I - \alpha \left(\sum_i w_i \tilde{P}^{(i)} \right), \quad \tilde{P}^{(i)} = U^T P^{(i)} U.$$

Error estimates

Key concept: *quasi-optimality*

$$\|x - \hat{x}\| \leq C \min_z \|x - Uz\|$$

where C can be controlled in some way.

Accuracy = Good space (consistency) + Quasi-optimality (stability)

Quasi-optimality

Define $\tilde{M} = W^T M U$ and $\Pi = U \tilde{M}^{-1} W^T M$:

$$x - Uy = (I - \Pi)x \qquad\qquad 0 = (I - \Pi)U$$

So we have the Galerkin error relation

$$x - Uy = (I - \Pi)(x - Uz)$$

for any candidate solution Uz. Take norms and minimize over z:

$$\|e\| \leq (1 + \kappa_G)\|e_{\min}\|$$

where

$$\kappa_G \equiv \|U\| \|\tilde{M}^{-1}\| \|W^T M\|.$$

Quasi-optimality

If $W^T U$ normalized so $W^T U = I$, then for linear parameterization

$$\|\tilde{M}^{-1}\| \leq \frac{1}{1 - \alpha \max_j \|\tilde{P}^{(j)}\|}$$

For 1-norm, have $\|M\|_1 \leq 1 + \alpha$, so if $\|\tilde{P}^{(j)}\|_1 < \alpha^{-1}$ for $j = 1, \ldots, d$,

$$\kappa_G \leq \frac{(1+\alpha)\|U\|_1 \|W\|_\infty}{1 - \alpha \max_j \|\tilde{P}^{(j)}\|_1}.$$

That is, we can bound the quasi-optimality constant *offline*.

Galerkin Shortcomings

For nonlinear parameterizations, still need

$$\tilde{M}(w) = U^T M(w) U.$$

Without a trick, have to

- Form all of $M(w)$
- Do k matrix-vector products with $M(w)$

Comparable to cost of standard PageRank algorithm!

DEIM Approach

DEIM = Discrete Empirical Interpolation Method

Goal:	$r = MUy - b \approx 0$ or $Uy \approx x$
DEIM ansatz:	minimize $\|r_{\mathcal{I}}\|$ for chosen indices $\mathcal{I}$, $|\mathcal{I}| \geq k$.

Only requires a few rows/columns of $M(w)$! But how to choose $\mathcal{I}$?

- More expensive if we choose high-degree nodes
  (Much more an issue in social networks than physical problems)
- What about accuracy? Choose "important" (high-PR) nodes?

Cost of forming the system

Typical case: $P = AD^{-1}$, given $A(w)$. Think of partitioning:

$$\begin{bmatrix} A_{11} & A_{12} & 0 \\ A_{21} & A_{22} & A_{23} \\ 0 & A_{32} & A_{33} \end{bmatrix}$$

If we enforce the first block equation, we need the colored blocks

$$\begin{bmatrix} \color{blue}{A_{11}} & \color{blue}{A_{12}} & 0 \\ \color{red}{A_{21}} & \color{red}{A_{22}} & A_{23} \\ 0 & \color{red}{A_{32}} & A_{33} \end{bmatrix}$$

where blue = used in DEIM equations, red = needed for normalization.

If $A = \sum_j w_j A^{(j)}$, no need to compute entries for normalization.

Cost of forming the system

Graph theoretic terms: if $A(w)$ is linear, cost to form $M_{\mathcal{I},:}U$ is

$$\sum_{v \in \mathcal{I}} \text{inDegree}(v)$$

Issue: Social networks have some very high-degree nodes!

Quasi-optimality

Analysis for DEIM $\approx$ analysis for Galerkin; in one-norm, have

$$\kappa_{\mathrm{DEIM}} \leq (1+\alpha)\|U\|_1 \|(M_{\mathcal{I},:}(w)U)^\dagger\|_1$$

Key: well-posedness of the projected least squares problem.

- Pro: Estimating $\|(M_{\mathcal{I},:}(w)U)^\dagger\|_1$ is cheap given $M_{\mathcal{I},:}(w)U = QR$.
- Con: A priori bounds are hard

Choosing the interpolation set

- Key: keep $M_{\mathcal{I},:}$ far from singular.
- If $|\mathcal{I}| = k$, this is a *subset selection* over rows of MU.
- Have standard techniques (e.g. pivoted QR)
- Want to pick $\mathcal{I}$ *once*, so look at rows of

$$Z = \begin{bmatrix} M(w^{(1)})U & M(w^{(2)})U & \dots \end{bmatrix}$$

for sample parameters $w^{(i)}$.

Application: Learning to Rank

Goal: Given $T = \{(i_q, j_q)\}_{q=1}^{|T|}$, find w that mostly ranks i_q over j_1.

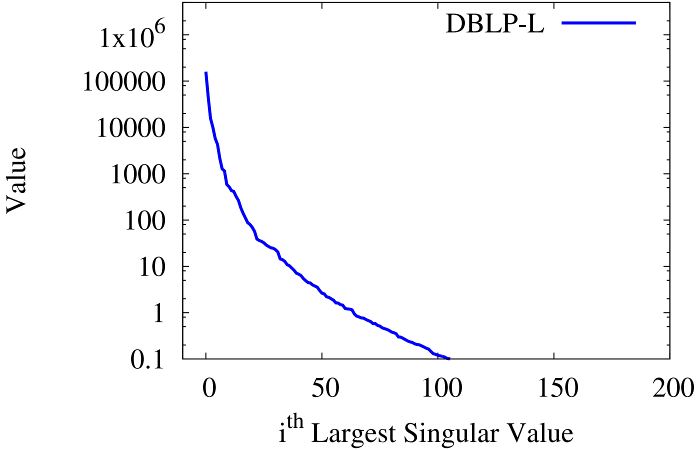
- Standard: Gradient descent on full problem
  - One PR computation for objective
  - One PR computation for each gradient component
  - Costs $d + 1$ PR computations per step
- With model reduction
  - Rephrase objective in reduced coordinate space
  - Use factorization to solve PR for objective
  - Re-use same factorization for gradient

Test case

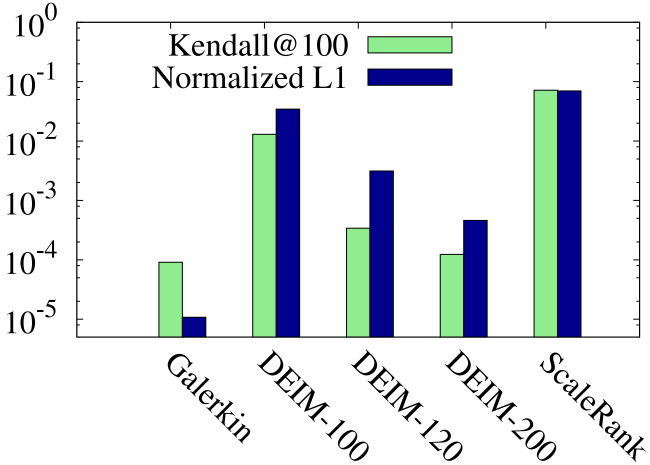
Test case: DBLP, 3.5M nodes, 18.5M edges, 7 params

- Goal: Learning to rank (8 papers for training)
- Consider linear parameterization (B-G and DEIM both apply)
- Compare to ScaleRank (more restrictive than we are, but applies here)
- This is a good case – see paper for some others

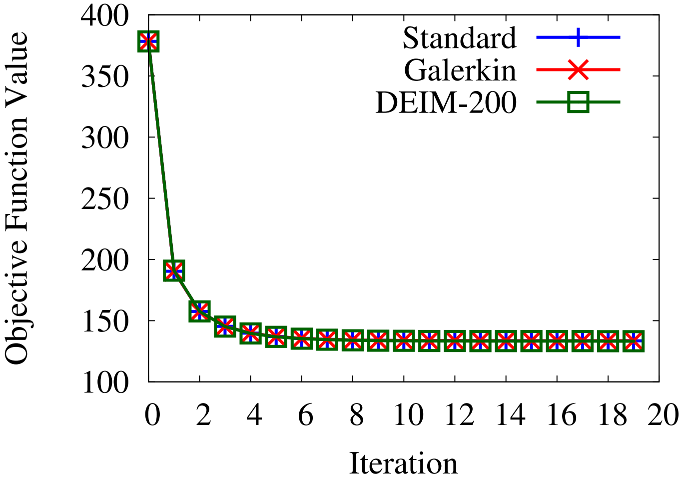
DBLP singular values



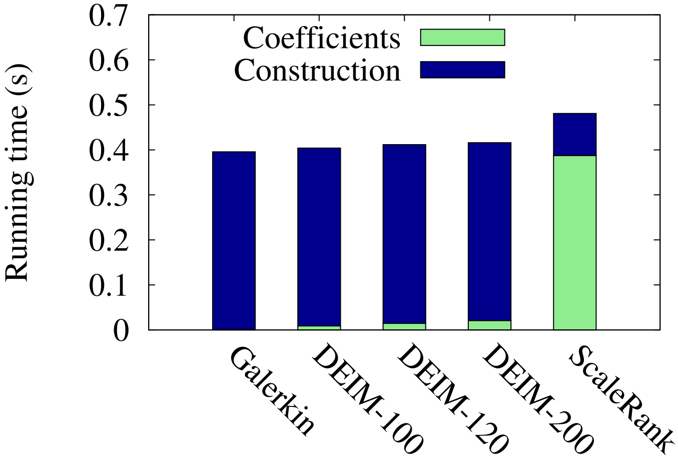
DBLP accuracy



DBLP learning task



DBLP running times (PR at *all* nodes)



The Punchline

Test case: DBLP, 3.5M nodes, 18.5M edges, 7 params

Cost per Iteration:

Method	Standard	Bubnov-Galerkin	DEIM-200
Time(sec)	159.3	0.002	0.033

Improvement of nearly *four or five orders of magnitude*.

For more

Edge-Weighted Personalized PageRank:
Breaking a Decade-Old Performance Barrier

Wenlei Xie, David Bindel, Johannes Gehrke, and Al Demers

Submitted to KDD