NYU Center for Data Science

Week 11.2: PageRank extensions

DS-GA 1004: Big Data

This week

- PageRank
  - [Page, Brin, Motwani, Winograd, 1999]
- Extensions to PageRank

Personalizing PageRank

- The uniform teleportation model isn't realistic
  - Jumping to ***any*** page? Really?

- If $e = [1, 1, 1, \ldots 1]$, then PageRank computes

$$\mathbf{p} = (a * \mathbf{M} + (1-a) * 1/N * ee^T)\mathbf{p}$$

Personalizing PageRank

- The uniform teleportation model isn't realistic
  - Jumping to ***any*** page? Really?
- If e = [1, 1, 1, … 1], then PageRank computes

$$\mathbf{p} = (a * \mathbf{M} + (1-a) * 1/N * ee^T)\mathbf{p}$$

- 1/N * e is the uniform distribution
  - what if we replaced it by something else?

Personalizing PageRank

- The uniform teleportation model isn't realistic
  - Jumping to ***any*** page? Really?
- If $e = [1, 1, 1, \ldots 1]$, then PageRank computes

$$\mathbf{p} = (a * \mathbf{M} + (1-a) * 1/N * ee^T)\mathbf{p}$$

This kills sparsity :(

- $1/N * e$ is the uniform distribution
  - what if we replaced it by something else?

Personalized PageRank

$$\mathbf{p} \quad = (a * \mathbf{M} + (1-a) * 1/N * ee^T)\mathbf{p}$$

Personalized PageRank

$$\mathbf{p} = (a * \mathbf{M} + (1-a) * 1/N * ee^T)\mathbf{p}$$

$$= a * \mathbf{M}\mathbf{p} + (1-a) * 1/N * ee^T\mathbf{p}$$

Personalized PageRank

$\mathbf{p} = (a * \mathbf{M} + (1-a) * 1/N * ee^T)\mathbf{p}$

$= a * \mathbf{Mp} + (1-a) * 1/N * ee^T\mathbf{p}$ ($e^T\mathbf{p} = 1$ because $\mathbf{p}$ is a distribution)

$= a * \mathbf{Mp} + (1-a) * 1/N * e$

Personalized PageRank

$$
\begin{aligned}
\mathbf{p} &= (a * \mathbf{M} + (1-a) * 1/N * ee^T)\mathbf{p} \\
&= a * \mathbf{Mp} + (1-a) * 1/N * ee^T\mathbf{p} && (e^T\mathbf{p} = 1 \text{ because } \mathbf{p} \text{ is a distribution}) \\
&= a * \mathbf{Mp} + (1-a) * 1/N * e && (\text{replace } 1/N * e \text{ by an arbitrary dist. } \mathbf{q}) \\
&= a * \mathbf{Mp} + (1-a) * \mathbf{q}
\end{aligned}
$$

- **q** is the *personalization vector* (distribution)
  - E.g., uniform over pages about dinosaurs

Personalized PageRank

$$\mathbf{p} = (a * \mathbf{M} + (1\text{-}a) * 1/N * ee^T)\mathbf{p}$$

$$= a * \mathbf{Mp} + (1\text{-}a) * 1/N * ee^T\mathbf{p}$$ ($e^T\mathbf{p} = 1$ because **p** is a distribution)

$$= a * \mathbf{Mp} + (1\text{-}a) * 1/N * e$$ (replace $1/N * e$ by an arbitrary dist. **q**)

$$= a * \mathbf{Mp} + (1\text{-}a) * \mathbf{q}$$

- **q** is the *personalization vector* (distribution)
  - E.g., uniform over pages about dinosaurs

We can still do power iteration here!

Even better: writing the update this way preserves sparsity in **M**

Distributed PageRank

```python
from graphframes.examples import Graphs
g = Graphs(sqlContext).friends()  # Get example graph

# Run PageRank until convergence to tolerance "tol".
results = g.pageRank(resetProbability=0.15, tol=0.01)
# Display resulting pageranks and final edge weights
# Note that the displayed pagerank may be truncated, e.g., missing the E notation.
# In Spark 1.5+, you can use show(truncate=False) to avoid truncation.
results.vertices.select("id", "pagerank").show()
```

- Core computation is **matrix multiplication**
  - This parallelizes very well
  - Complexity depends on network sparsity

$$\mathbf{p} \leftarrow a * \mathbf{M}\mathbf{p} + (1-a) * \mathbf{q}$$

- Also possible in Spark using the GraphX package

- High-level interface: GraphFrames
  - *"GraphX is to RDDs as GraphFrames are to DataFrames."*

Strategies for fighting link spam (1)

- **TrustRank**: bias **q** by **human intervention** and **curation**

- Elevated trust for…
    - Certain sites (e.g., cdc.gov/coronavirus, or high PageRank sites)
    - Domains (nyu.edu)
    - Top-level domains (.edu >> .biz)

- Some curation is almost certainly necessary, but can border on censorship
    - This can be easy to mess up! What if we include .edu, but forget .ac.uk?

Strategies for fighting link spam (2)

- SpamMass(u) = (PageRank(u) - TrustRank(u)) / PageRank(u)
    - Large values = probably **spam**
    - Small values = probably **not spam**
    - But really just measuring how much linkage comes from “trusted” sites
- When searching, drop any page with SpamMass(u) > **threshold**

PageRank for recommendation?

- "Personalization vector" **q** = distribution over past items

  What's the network between items?

- Social network recommendation

  ⇒ friends / followers

- General user/items setting

  ⇒ Link item → item if they have (many) users in common?

Wrap-up

- PageRank provides a flexible framework for exploiting network topology in search
- Most extensions are designed to combat “link spam”
- Core computation is a linear algebra problem (eigenvector) that parallelizes well.