Reduce and Aggregate: Similarity Ranking in Multi-Categorical Bipartite Graphs

Alessandro Epasto

J. Feldman*, S. Lattanzi*, S. Leonardi°, V. Mirrokni*.

*Google Research °Sapienza U. Rome
Motivation

- Recommendation Systems:
  - Bipartite graphs with Users and Items.
  - Identify similar users and suggest relevant items.
  - Concrete example: The AdWords case.

- Two key observations:
  - Items belong to different categories.
  - Graphs are often lopsided.
Modeling the Data as a Bipartite Graph

- Nike Store New York
- Soccer Shoes
- Soccer Ball
- Retailers
- Apparel
- Sport Equipment
- Millions of Advertisers
- Billions of Queries
Personalized PageRank

For a node $v$ (the seed) and a probability alpha $\alpha$,

The stationary distribution assigns a similarity score to each node in the graph w.r.t. node $v$. 
The Problem

Millions of Advertisers

Billions of Queries

Retailers
-Nike Store New York

Apparel
-Soccer Shoes

Sport Equipment
-Soccer Ball

Hundreds of Labels

Millions of Advertisers

Billions of Queries
Other Applications

- General approach applicable to several contexts:
  - User, Movies, Genres: find similar users and suggest movies.
  - Authors, Papers, Conferences: find related authors and suggest papers to read.
Semi-Formal Problem Definition

Advertisers

Queries
Semi-Formal Problem Definition

Advertisers

Queries
Semi-Formal Problem Definition

Advertisers

Queries

Labels:  

Semi-Formal Problem Definition

Advertisers

Queries

Labels:

Goal: Find the nodes most “similar” to A.
How to Define Similarity?

- We address the computation of several node similarity measures:
  - Neighborhood based: Common neighbors, Jaccard Coefficient, Adamic-Adar.
  - Paths based: Katz.

- Experimental question: which measure is useful?
- Algorithmic questions:
  - Can it scale to huge graphs?
  - Can we compute it in real-time?
Our Contribution

- **Reduce and Aggregate**: general approach to induce real-time similarity rankings in multi-categorical bipartite graphs, that we apply to several similarity measures.

- Theoretical guarantees for the precision of the algorithms.

- Experimental evaluation with real world data.
Personalized PageRank

For a node \( v \) (the seed) and a probability \( \alpha \)

The stationary distribution assigns a similarity score to each node in the graph w.r.t. node \( v \).
Challenges

- Our graphs are too big (**billions** of nodes) even for very large-scale MapReduce systems.
- MapReduce is not real-time.
- We cannot pre-compute the rankings for each subset of labels.
Reduce and Aggregate

**Reduce**: Given the bipartite and a category construct a graph with only $A$ nodes that preserves the ranking on the entire graph.

**Aggregate**: Given a node $v$ in $A$ and the reduced graphs of the subset of categories interested determine the ranking for $v$. 
Reduce (Precomputation)

Advertisers

Queries
Reduce (Precomputation)

Advertisers

Queries

Precomputed Rankings
Reduce (Precomputation)

Advertisers

Queries

Precomputed Rankings

Precomputed Rankings
Reduce (Precomputation)

Advertisers

Queries

Precomputed Rankings

Precomputed Rankings

Precomputed Rankings
Aggregate (Run Time)

Precomputed Rankings + Precomputed Rankings = Ranking of Red + Yellow
Reduce for Personalized PageRank

- Markov Chain state aggregation theory (Simon and Ado, ’61; Meyer ’89, etc.).
- 750x reduction in the number of node while preserving correctly the PPR distribution on the entire graph.
Run-time Aggregation
Step 1: Partition the Markov chain into DISJOINT subsets
Koury et al. Aggregation-Disaggregation Algorithm

Step 2: Approximate the stationary distribution on each subset independently.
Koury et al. Aggregation-Disaggregation Algorithm

Step 3: Consider the transition between subsets.
Koury et al. Aggregation-Disaggregation Algorithm

Step 4: Aggregate the distributions. Repeat until convergence.
Aggregation in PPR

Precompute the stationary distributions individually
Precompute the stationary distributions individually.
Aggregation in PPR

The two subsets are not disjoint!
Our Approach

- The algorithm is based **only** on the reduced graphs with Advertiser-Side nodes.

- The aggregation algorithm is scalable and converges to the correct distribution.
Experimental Evaluation

- We experimented with publicly available and proprietary datasets:
  - Query-Ads graph from Google AdWords > 1.5 billions nodes, > 5 billions edges.
  - DBLP Author-Papers and Patent Inventor-Inventions graphs.
  - Ground-Truth clusters of competitors in Google AdWords.
Patent Graph

Precision vs Recall

- Inter
- Jaccard
- Adamic-Adar
- Katz
- PPR
Conclusions and Future Work

- It is possible to compute several similarity scores on very large bipartite graphs in real-time with good accuracy.

- Future work could focus on the case where categories are not disjoint is relevant.
Thank you for your attention
Reduction to the Query Side
Reduction to the Query Side

This is the larger side of the graph.
Convergence after One Iteration

Kendall-Tau Correlation

Position (k)

Kendall-Tau

DBLP
Patent
Query-Ads (cost)
Convergence

Approximation Error vs # Iterations

1-Cosine Similarity

1-Cosine

DBLP (1 - Cosine)  Patent (1 - Cosine)

Iterations