Pagerank:

Page ranking pages and Beyond

Alexander Munoz

28 March 2017

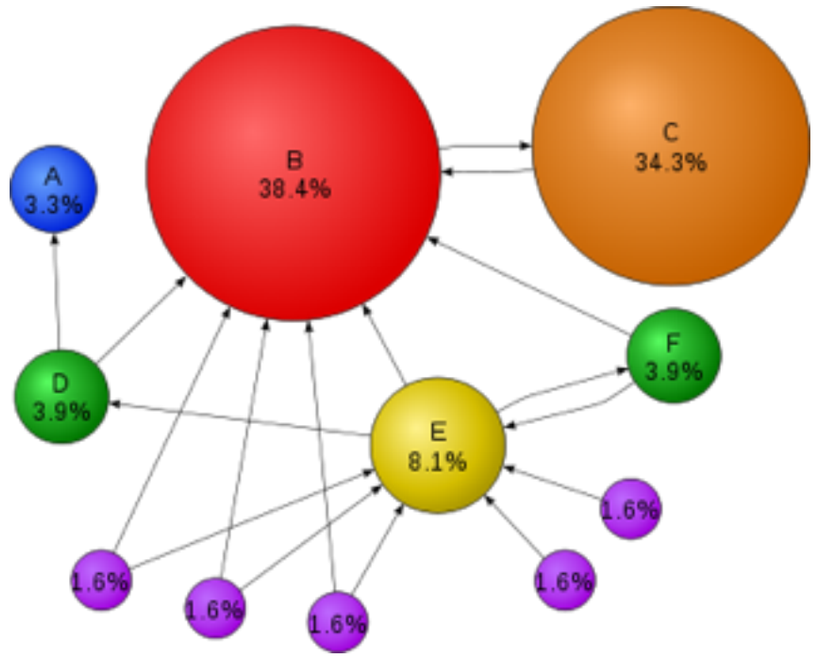
Algorithms Interest Group

Outline

- High-level description
- Low-level Description
- Examples
- Google's Synthesis
- Applications

High-Level

- Pagerank solves a system of “score” equations
- Yields a probability distribution that a person randomly clicking links will arrive at a particular page



High-Level

- Google interprets a link from page A to page B as a vote by page A for page B

- However, not all votes are equal

- The rank (importance) of a webpage gets factored in — high ranked votes weigh more heavily

Random-Surfer Model

- Probability that a random surfer clicks on a link is given by the number of links on a page
- The probability of reaching a page is the sum of probabilities for the surfer followings links to the page
- Introduce a damping factor which gives a chance to jump to another page at random — minimum Pagerank

Lower-Level

- Within a network, we can calculate the Pagerank of a particular page

- Say page A has pages $T_1 \ldots T_n$ pointing to it and we have links going out of page A classified as C(A):

$$PR(A) = (1-d) + d\left[\frac{PR(T_1)}{C(T_1)} + \cdots + \frac{PR(T_n)}{C(T_n)}\right]$$

Lower-Level

- PR can be calculated using a simple iterative algorithm

- PR corresponds to the principal eigenvector of the normalized link matrix — we can calculate PR without knowing the final PR values of other pages

- Computation can be done iteratively or algebraically — Power method

Lower-Level

- Iterative: $PR(p_i, 0) = \frac{1}{N}$

$$PR(p_i, t+1) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j, t)}{L(p_j)}$$

$$R(t+1) = d\mathcal{M}R(t) + \frac{1-d}{N}1$$

where $\mathcal{M} = \frac{1}{L(p_j)}$

Converges when:

$$|R(t+1) - R(t)| < \epsilon$$

Lower-Level

- Algebraically:
  as t goes to infinity

$$R = d\mathcal{M}R + \frac{1-d}{N}\hat{1}$$

The solution is given by

$$R = (\mathbb{I} - d\mathcal{M})^{-1}\frac{1-d}{N}\hat{1}$$

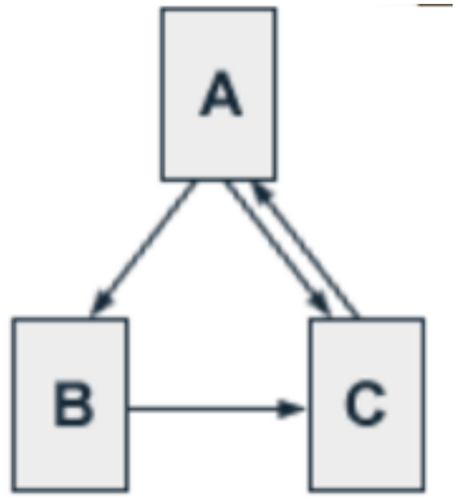
Lower-Level

- The previous calculations yield the same Pageranks if their results are normalized:

$$R_{power} = \frac{R_{iter}}{|R_{iter}|} = \frac{R_{alg}}{|R_{alg}|}$$

Lower-Level

- Quick demonstration
  PR(A)=0.5+0.5*PR(C)
  PR(B) = 0.5+0.5*(PR(A)/2)
  PR(C) = 0.5+0.5(PR(A)/2+PR(B))



Iteration	PR(A)	PR(B)	PR(C)
0	1	1	1
1	1	0.75	1.125
2	1.0625	0.765625	1.1484375
3	1.07421875	0.76855469	1.15283203
4	1.07641602	0.76910400	1.15365601
5	1.07682800	0.76920700	1.15381050
6	1.07690525	0.76922631	1.15383947
7	1.07691973	0.76922993	1.15384490
8	1.07692245	0.76923061	1.15384592
9	1.07692296	0.76923074	1.15384611
10	1.07692305	0.76923076	1.15384615
11	1.07692307	0.76923077	1.15384615
12	1.07692308	0.76923077	1.15384615

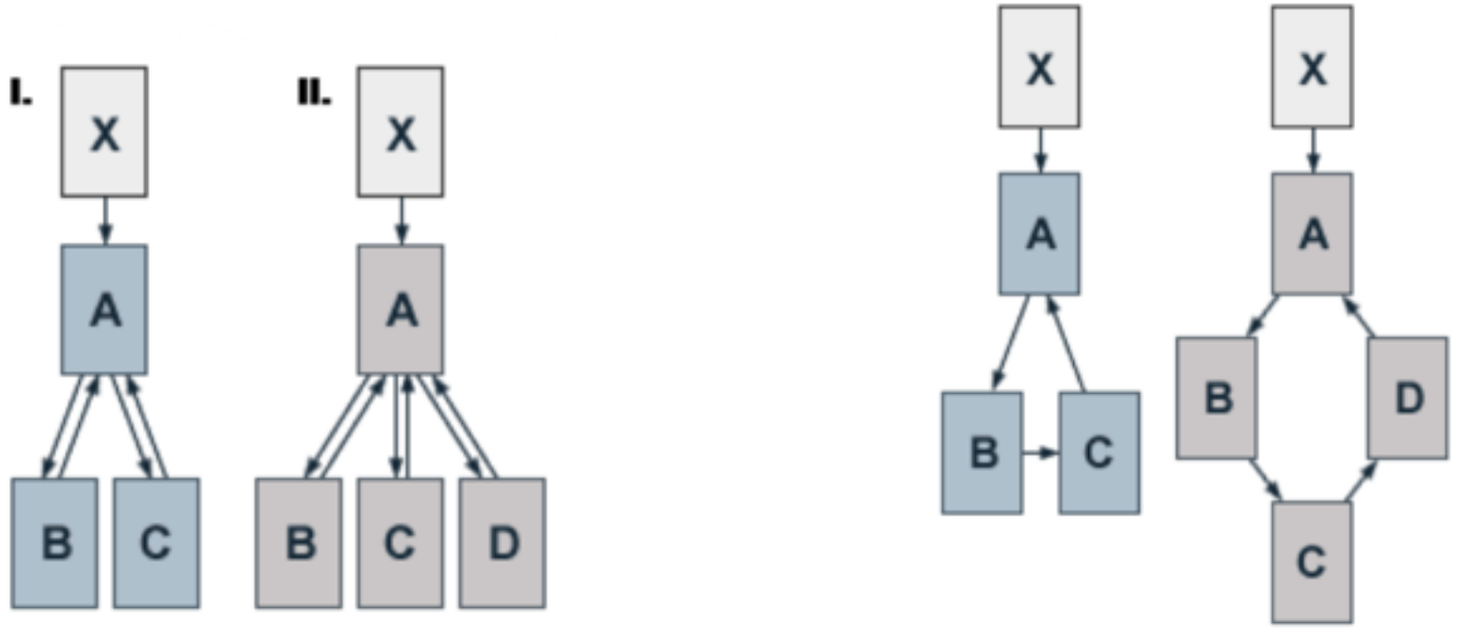
Improving your Pagerank

- Add new pages to your website — in a semi-intelligent way
- Swap links with websites which have high Pageranks
- Raise the number of inbound links (Advertising)

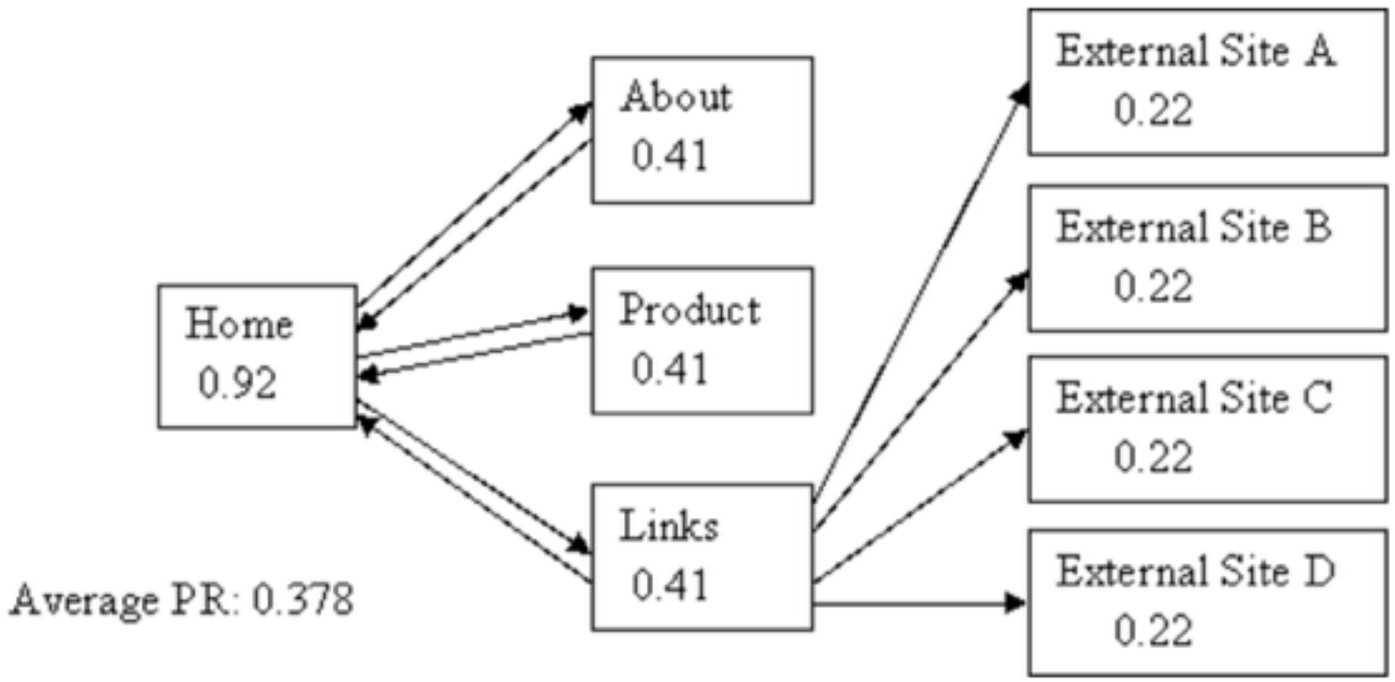
Improving your Pagerank

- When you add a new page to your site, link it to the front page

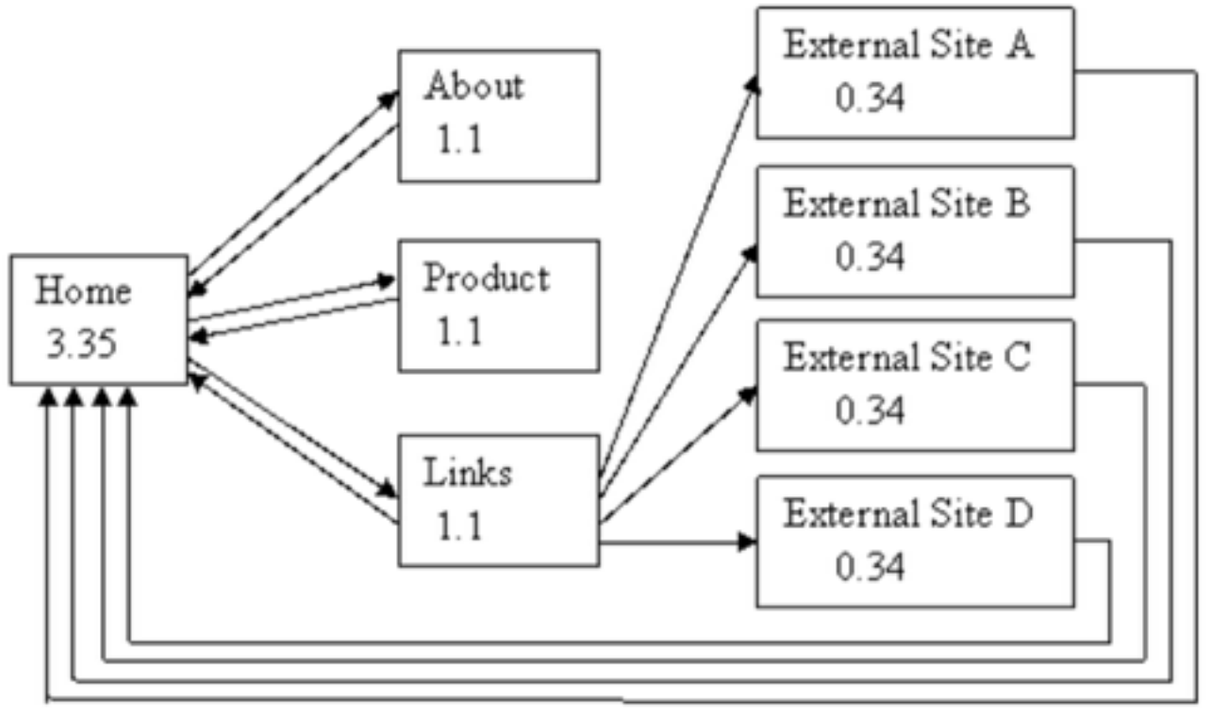
- You can reduce your front page's Pagerank by making circular references in your website



Improving your Pagerank



Average PR: 0.378



Average PR: 1.000

Improving your Pagerank

These manipulations are not enough — create good content instead

Google's Synthesis

- Ranking of webpages in Google was determined by three factors
  -Page specific factors
  -Anchor text of inbound links
  -Pagerank

- Measuring an inbound link's potential for pointing the correct information

"Calculating derivatives in **three dimensions**"

vs.

"**Calculating derivatives** in three dimensions"

Google's Synthesis

- Specific factor examples
  -Domain registration length
  -Penalize WhoIs Owner — spammers get punished
  -Keyword in title tag
  -Keyword density
  -Page loading speed via HTML
  -Outbound link theme
  -Reading level

- Many, many more factors: social signals, domain factors, page factors, algorithm rules, backlink factors…

Google's Synthesis

- In order to provide search results, Google computes an IR score from the first two components
- Pagerank multiplied with the IR score yields the general importance of the page

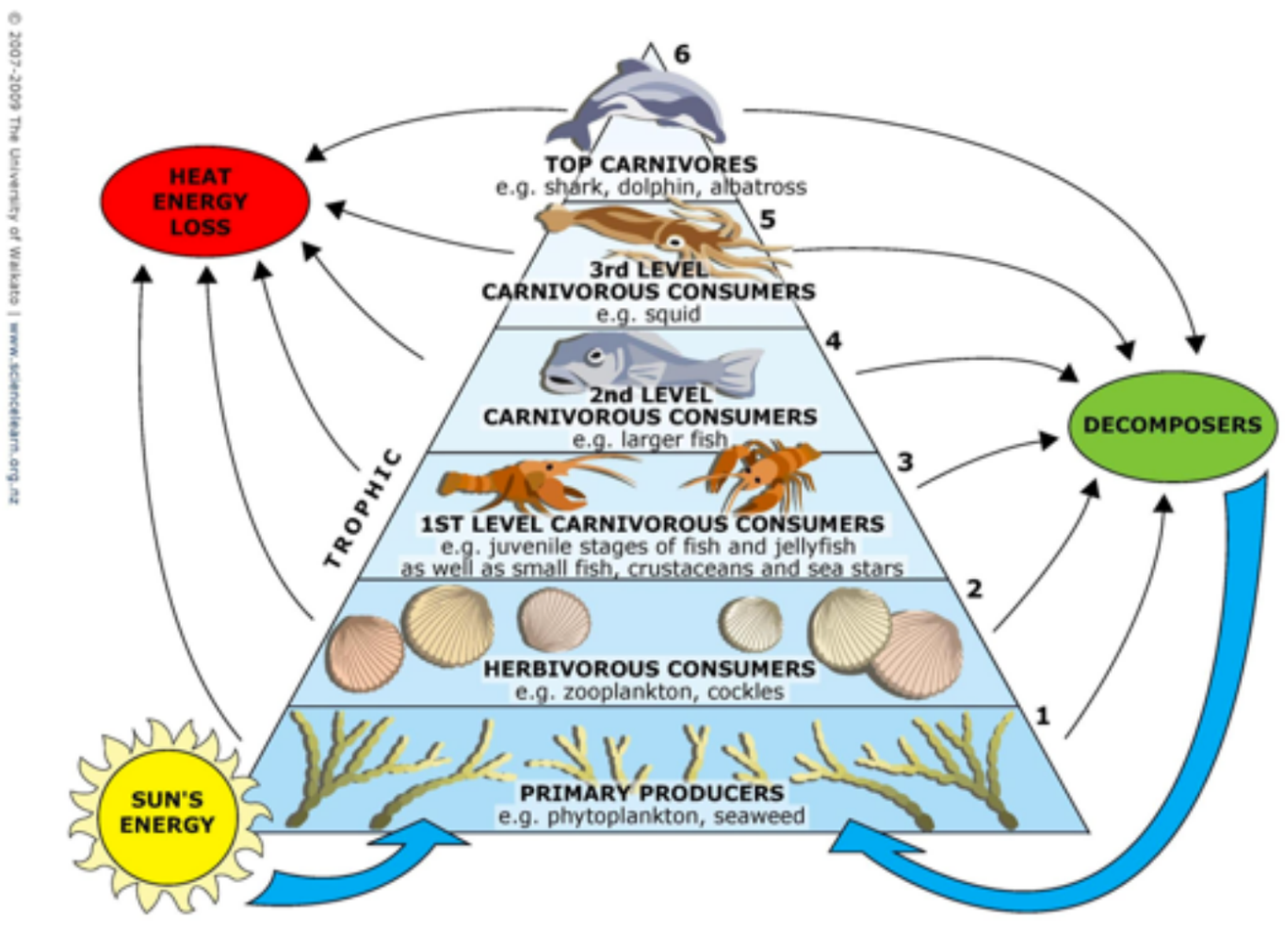
Google's Synthesis

Order	Google	Yahoo!	MSN
1	www.starbucks.com (⋄)	www.gevalia.com (⋄)	www.peets.com (∗)
2	www.coffeereview.com (†)	en.wikipedia.org/wiki/Coffee (△)	en.wikipedia.org/wiki/Coffee (△)
3	www.peets.com (∗)	www.nationalgeographic.com/coffee	www.coffeegeek.com (∗)
4	www.coffeegeek.com (∗)	www.peets.com (∗)	coffeetea.about.com (△)
5	www.coffeeuniverse.com (†)	www.starbucks.com (⋄)	coffeebean.com
6	www.coffeescience.org	www.coffeegeek.com (∗)	www.coffeereview.com (†)
7	www.gevalia.com (⋄)	coffeetea.about.com (△)	www.coffeeuniverse.com (†)
8	www.coffeebreakarcade.com	kaffee.netfirms.com/Coffee	www.tmcm.com
9	https://www.dunkindonuts.com	www.strong-enough.net/coffee	www.coffeeforums.com
10	www.cariboucoffee.com	www.cl.cam.ac.uk/coffee/coffee.html	www.communitycoffee.com
Approximate Number of Results:	447,000,000	151,000,000	46,850,246

Shared results for Google, Yahoo!, and MSN (∗); Google and Yahoo! (⋄); Google and MSN (†); and Yahoo! and MSN (△)

Applications

- Ecology — Food Webs
- Uses cyclical elements — Animal to detritus to plants to Animal
- How does the loss of a species cascade? Measure the importance of the species



Applications

- Recommendation Systems — e.g. Netflix

- User identifies what they like

- A movie is *relevant* for me if other *similar* people liked it
  and
  A person is *similar* to me if they like movies that are *relevant* to me

$$\begin{aligned}
\text{rel}(m) &= \sum_{\text{individuals } i \text{ who liked } m} \frac{\text{sim}(i)}{\text{number of movies } i \text{ liked}} \\
\text{sim}(i) &= \sum_{\text{movies } m \text{ which } i \text{ liked}} \frac{\text{rel}(m)}{\text{number of people who liked } m}
\end{aligned}$$

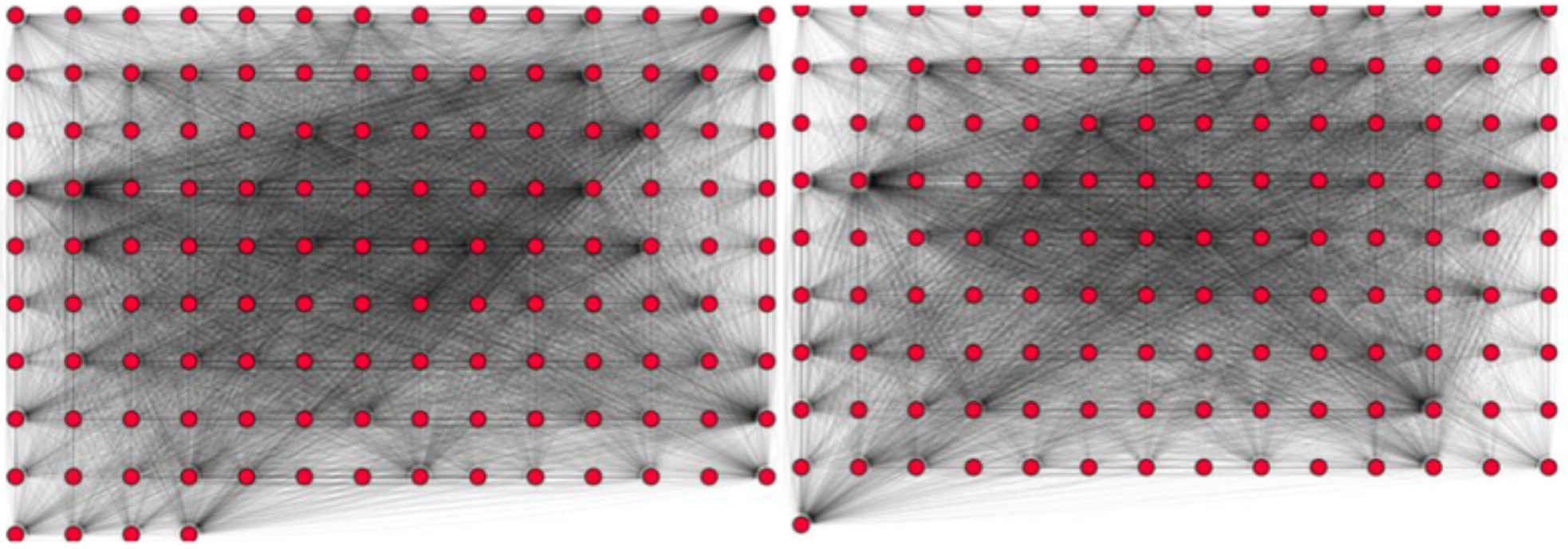
- Whenever user u likes product m, we draw two edges, one from node u to m and the other one from node m to u

Applications

- League of Legends Balance Analysis

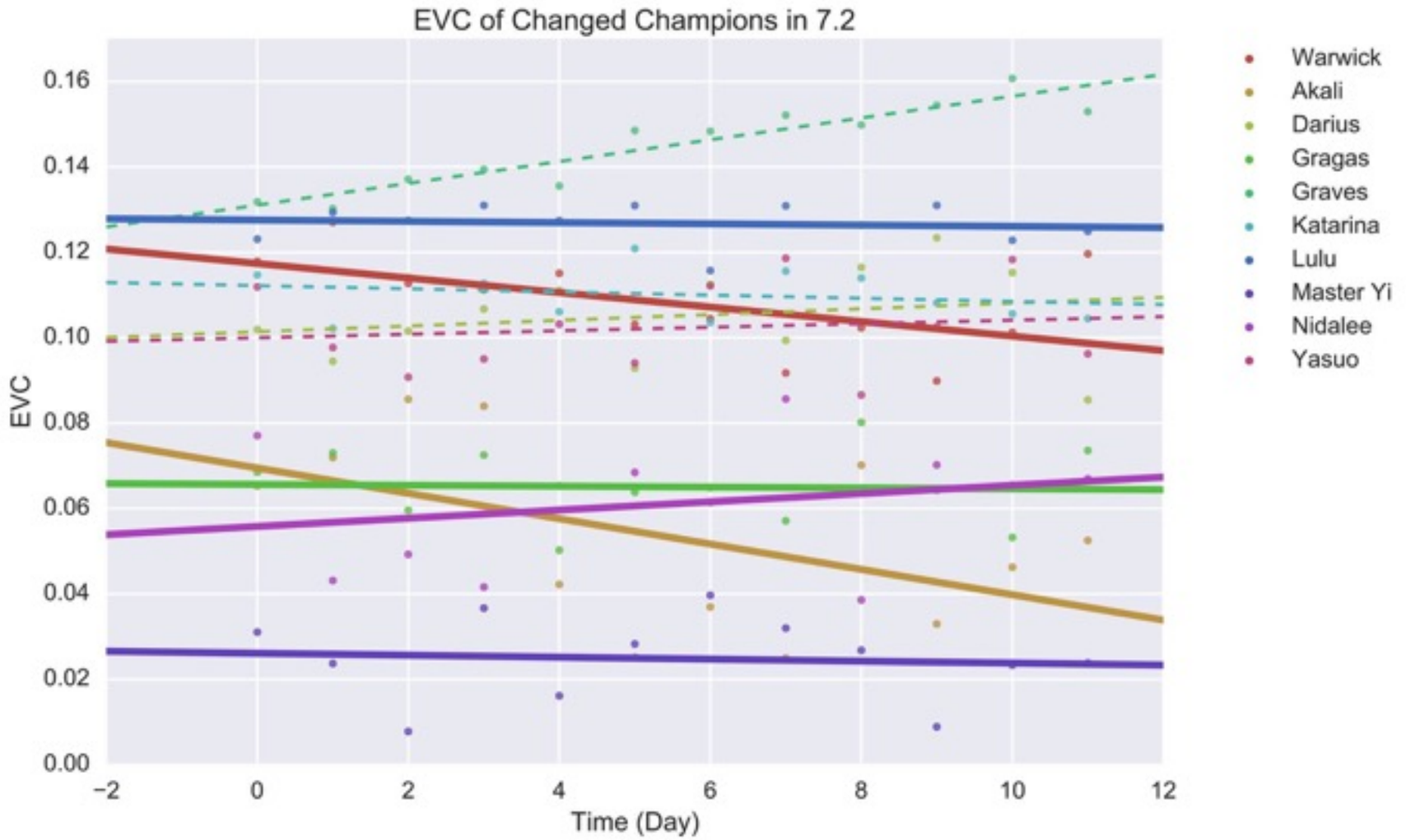
$$w_{i\to j} = (\text{popularity})\times(\text{winrate})$$

$$w_{i\to j} = \frac{\text{matches with } i \text{ vs } j}{\text{total matches}} \times \frac{\text{number of matches where } i \text{ beat } j}{\text{matches with } i \text{ vs } j} = \frac{N_{i\to j}}{M}.$$



Applications

- League of Legends Balance Analysis



Conclusion

- Pagerank is a simple algorithm which gives rise to a fair amount of complexity

- Pagerank-type algorithms have developed to build descriptions of a wide range of phenomena

Bibliography

- https://en.wikipedia.org/wiki/PageRank

- Examples and Principles: http://www.cs.princeton.edu/~chazelle/courses/BIB/pagerank.htm

- Larry Page: http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf

- Google specifics: https://prchecker.net/how-pagerank-is-used-in-google-search-engine-application.html

- Application: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1000494