

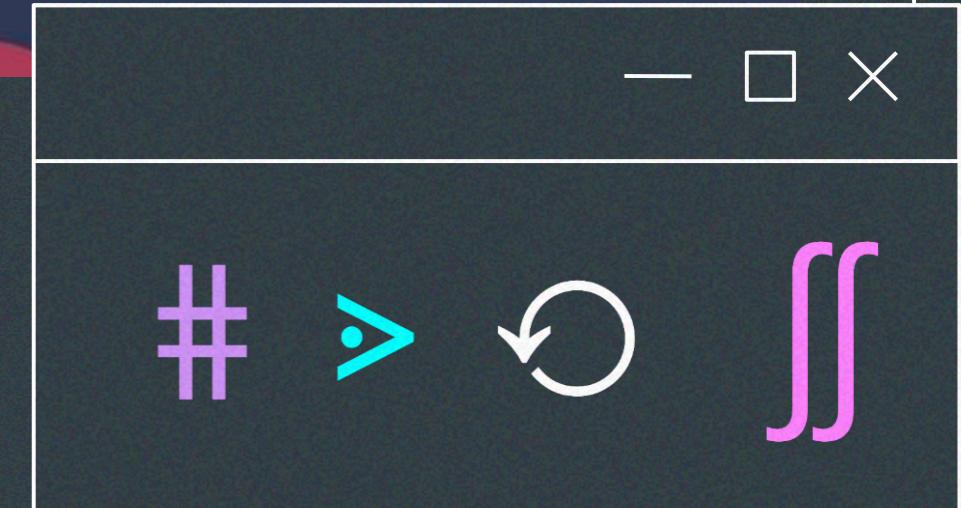
Google Pagerank

loganix



Introducere

Înțelegerea algoritmului Page Rank este crucială pentru succesul online. Succesul extraordinar și dominația motorului Google se datorează în principal algoritmului PageRank, care exploatează structura linkurilor din WWW pentru a determina un indice de popularitate al fiecărei pagini, independent de interogarea formulată de utilizator.



Ce este Page Rank?

Page Rank este un algoritm de analiză a hiperlegăturilor din Internet, folosit de motorul de căutare Google pentru a acorda o pondere fiecărui element dintr-o mulțime de documente interconectate prin hiperlegături, cu scopul măsurării importanței relative în cadrul mulțimii.

Dacă pagina A conține un link (o legătură) către pagina B, se presupune implicit că A afirmă despre B că acesta este important, deci B trebuie să fie mai bine cotat în clasamente.

Cu cât există mai multe legături calitative către un site, cu atât PageRank-ul acestuia va fi mai mare și locul în clasament mai înalt. Coeficientul PageRank este un număr întreg care poate lua valori între 0 și 10.



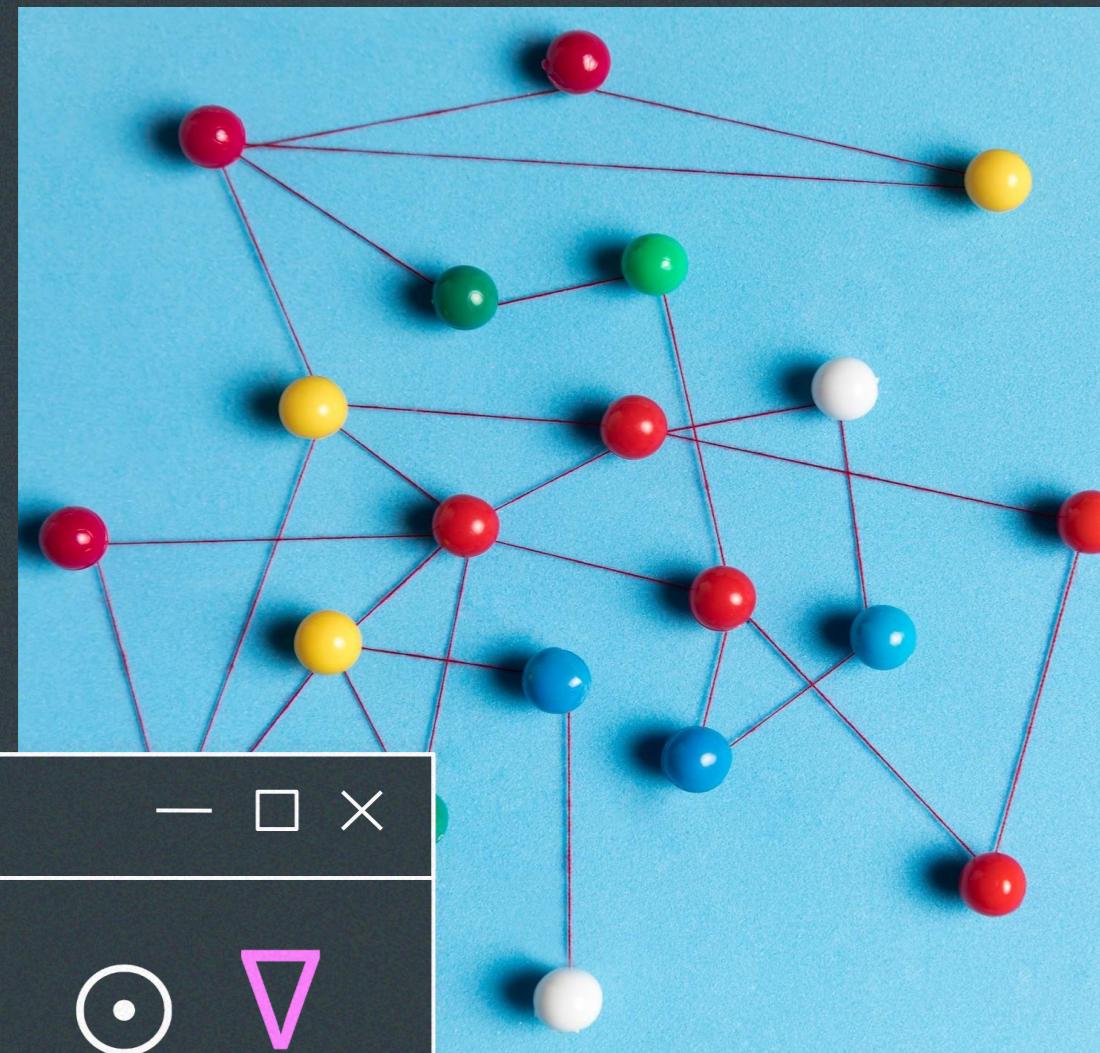
Evoluția Page Rank

Algoritmul Page Rank a fost dezvoltat de Larry Page și Sergey Brin la Google în 1996.

A revoluționat căutarea, dând prioritate backlink-urilor de calitate față de densitatea cuvintelor cheie.



Componente de Bază



Page Rank se bazează pe conceptul de analiză a linkurilor. Documentele de pe WEB (paginile WEB) sunt identificate de aplicațiile software ale motorului, numite roboți sau crawlere. Documentele sunt apoi indexate. Modulul de indexare extrage cuvintele cheie, constituind aşa numitul sac de cuvinte. Un alt modul, numit query module (modulul de interogare), convertește cererea formulată de utilizator, în limbaj natural, într-un vector cerere, cu care consultă indexul de conținut și extrage paginile relevante cererii. Modulul de ierarhizare ordonează descrescător aceste pagini în funcție de coeficienții de popularitate.

PageRank-ul este un vector ale cărui coordonate sunt coeficienții de popularitate ai paginilor WEB identificate de crawler. Acest vector este distribuția de echilibru a unui lanț Markov definit pe graful WEB.

Să definim mai întâi lanțul Markov ce stă la baza algoritmului PageRank. Considerăm $W = \{1, 2, \dots, m\}$ mulțimea tuturor paginilor WEB, $H = (h_{ij})$ matricea de conectivitate a lui W sau matricea hyperlink:

$$h_{ij} = \begin{cases} 1, & \text{dacă există link în pagina } i \text{ către pagina } j, \\ 0, & \text{dacă nu există link în pagina } i \text{ către pagina } j. \end{cases}$$

Suma elementelor de pe linia i a matricei H indică numărul de out-linkuri, adică numărul de linkuri din pagina i către alte pagini sau ea însăși.

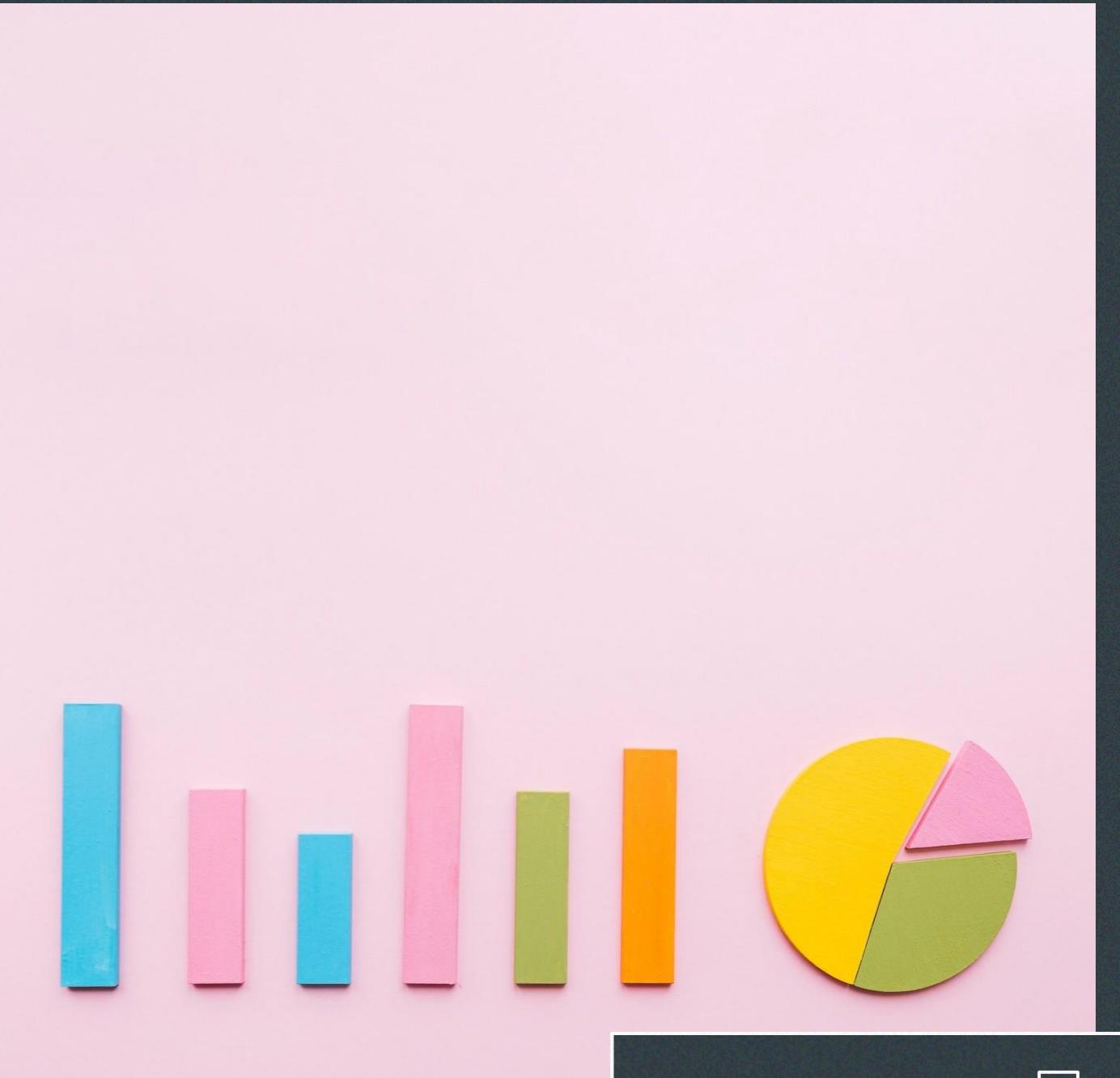
$$r_i = \sum_{j=1}^m h_{ij}. r_i \text{ se numește ordinul ieșirilor din pagina } i.$$

$$p_{ij} = \begin{cases} \frac{1}{r_i}, & \text{dacă există link în pagina } i \text{ către pagina } j, \\ 0, & \text{dacă nu există link în pagina } i \text{ către pagina } j. \end{cases}$$

Page Rank vs. Densitatea Cuvintelor Cheie

În timp ce densitatea cuvintelor cheie este importantă, Page Rank pune accentul pe importanța backlink-urilor de calitate.

Echilibrarea ambelor este crucială pentru optimizarea motoarelor de căutare.



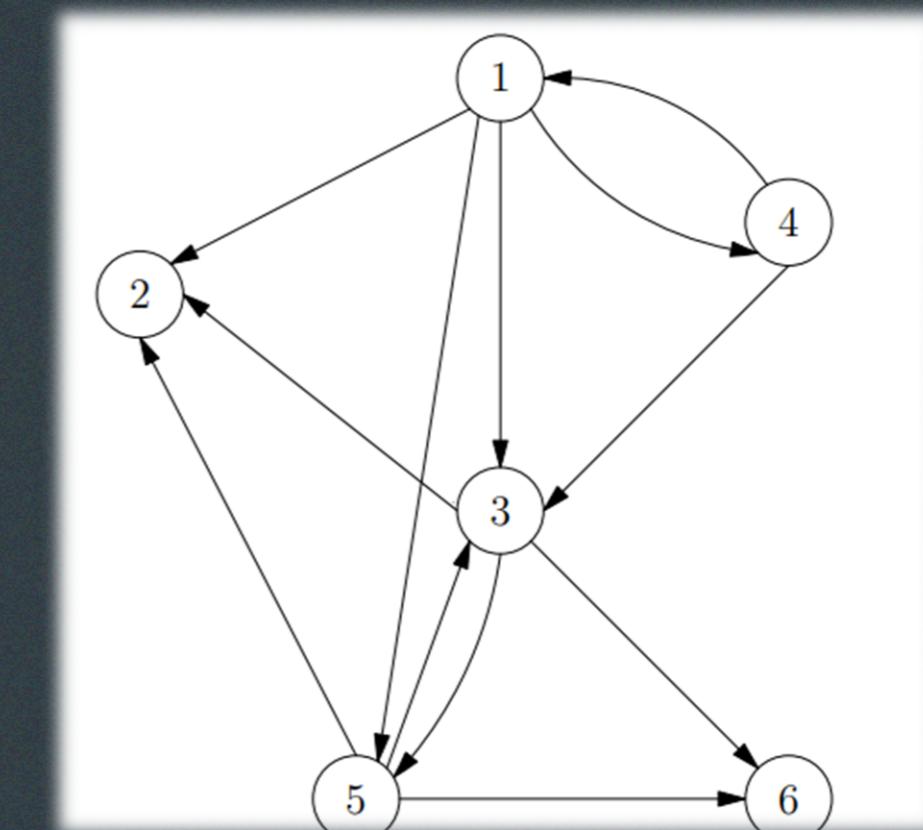
Long Story Short

În esență, Google interpretează un link de la pagina A la pagina B ca un vot, de la pagina A, pentru pagina B.

Dar aceste voturi nu cântăresc la fel, pentru că Google analizează și pagina care votează.

$$H = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

De exemplu ordinul de ieșire din pagina 5 este $r_5 = 3$, deci probabilitatea de a trece din pagina 5 în oricare din paginile $\{1, 2, \dots, 6\}$ este $p_{5j} = h_{5j}/3$. Altfel spus, cu aceeași probabilitate de $1/3$ un surfer poate trece din pagina 5 în pagina 2, 3 sau 6.



Originalul Algoritm Page Rank

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

Unde:

- $PR(A)$ este rangul paginii A,
- $PR(T_i)$ este rangul paginii T_i care trimite către pagina A,
- $C(T_i)$ este numărul de link-uri de ieșire din pagina T_i ,
- d este un factor de amortizare care poate lua o valoare între 0 și 1

$$\text{PR}(T_1)/C(T_1) + \dots + \text{PR}(T_n)/C(T_n)$$

PageRank-ul al paginilor Ti care leagă la pagina A nu influențează uniform PageRank al paginii A.

PageRank al unei pagini T este întotdeauna ponderat cu numărul de legături de ieșire C(T) de pe pagina T.

Ceea ce înseamnă că cu cât mai multe legături de ieșire are o pagină T, cu atât pagina A va beneficia mai puțin de un link către ea pe pagina T.

Se adună apoi PageRank-ul ponderat al paginilor Ti. Rezultatul este că un link de intrare suplimentar pentru pagina A va crește întotdeauna PageRank™ al paginii A.

$$\text{PR}(A) = (1-d) + d * \left(\frac{\text{PR}(T_1)}{C(T_1)} + \dots + \frac{\text{PR}(T_n)}{C(T_n)} \right)$$

La urma urmei, suma PageRank-urilor ponderate ale tuturor paginilor Ti este înmulțită cu un factor de amortizare d care poate fi setat între 0 și 1.

Astfel, extinderea beneficiului PageRank pentru o pagină cu o altă pagină care leagă la aceasta este redusă.

Deci, probabilitatea ca utilizatorul aleatoriu să ajungă la o pagină este suma probabilităților pentru respectivul utilizator care urmează link-urile către această pagină.



Factorul de Amortizare d

Probabilitatea ca surferul aleatoriu să nu se oprească să facă click pe linkuri este dată de factorul de amortizare d , care depinde de probabilitate, prin urmare, este setat între 0 și 1.

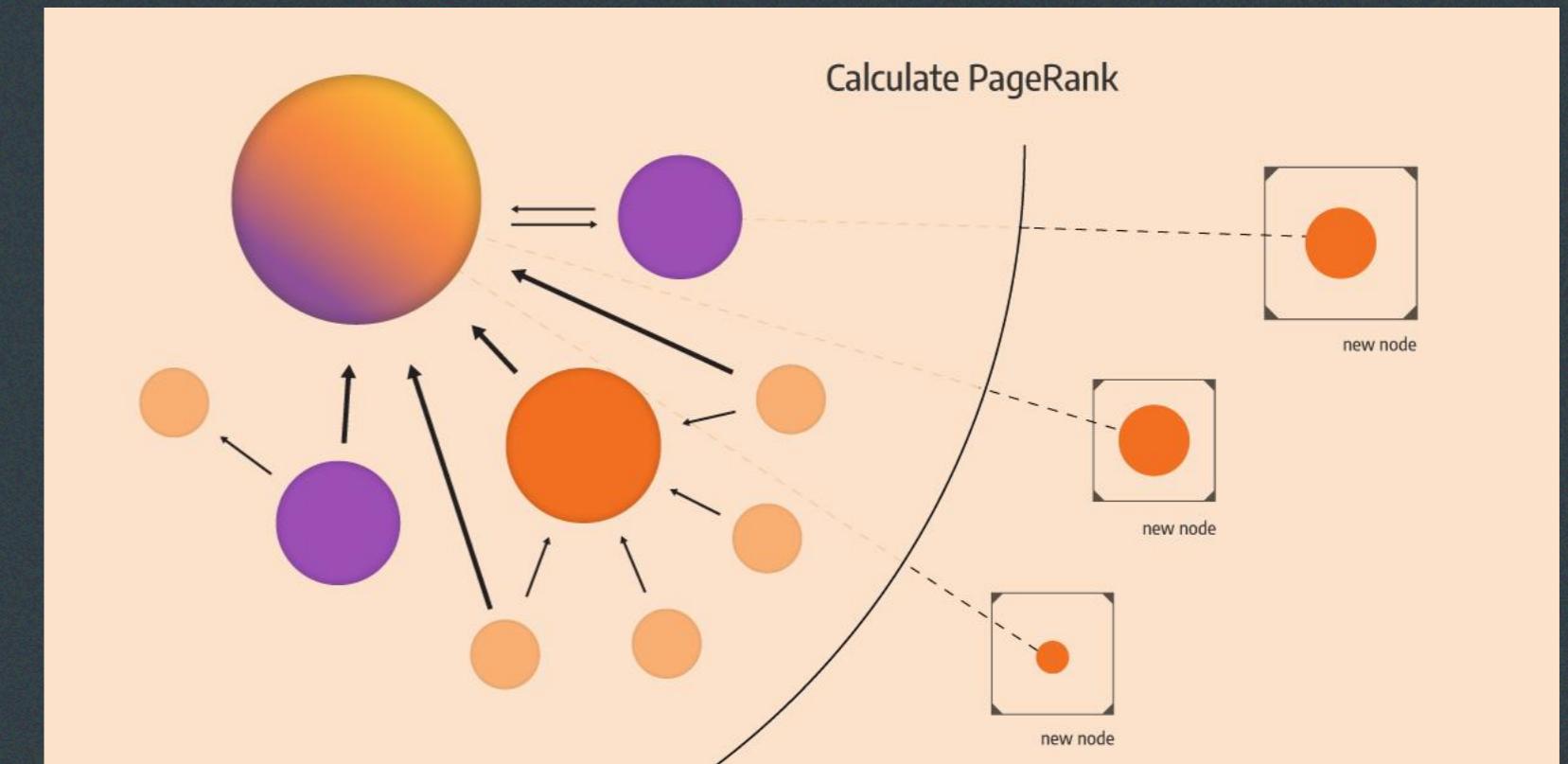
Cu cât d este mai mare, cu atât este mai probabil ca surferul aleatoriu să continue să facă click pe linkuri.

Deoarece surferul sare la o altă pagină la întâmplare după ce a încetat să mai facă click pe linkuri, probabilitatea este implementată ca o constantă $(1-d)$ în algoritm.

Indiferent de legăturile de intrare, probabilitatea ca surferul aleatoriu să sară la o pagină este întotdeauna $(1-d)$, astfel încât o pagină are întotdeauna un PageRank minim.

Recursivitatea

Este evident că algoritmul PageRank nu clasifică întregul site web, dar este determinat pentru fiecare pagină în parte.



În plus, PageRank al paginii A este definit recursiv de PageRank al acelor pagini care se leagă la pagina A.

O Notație Diferită a Algoritmului PageRank



$$PR(A) = (1-d) / N + d (PR(T_1)/C(T_1) + \dots + PR(T_n)/C(T_n))$$

unde N este numărul total al tuturor paginilor de pe web.

Unde:

d este factorul de amortizare (damping factor), de obicei setat la 0.85, care reflectă probabilitatea ca un utilizator să continue să navigheze de la o pagină la alta.

N este numărul total de pagini.

P_i reprezintă paginile care link-ează către pagina P .

$L(P_i)$ este numărul de link-uri de ieșire pe care le are pagina P_i .

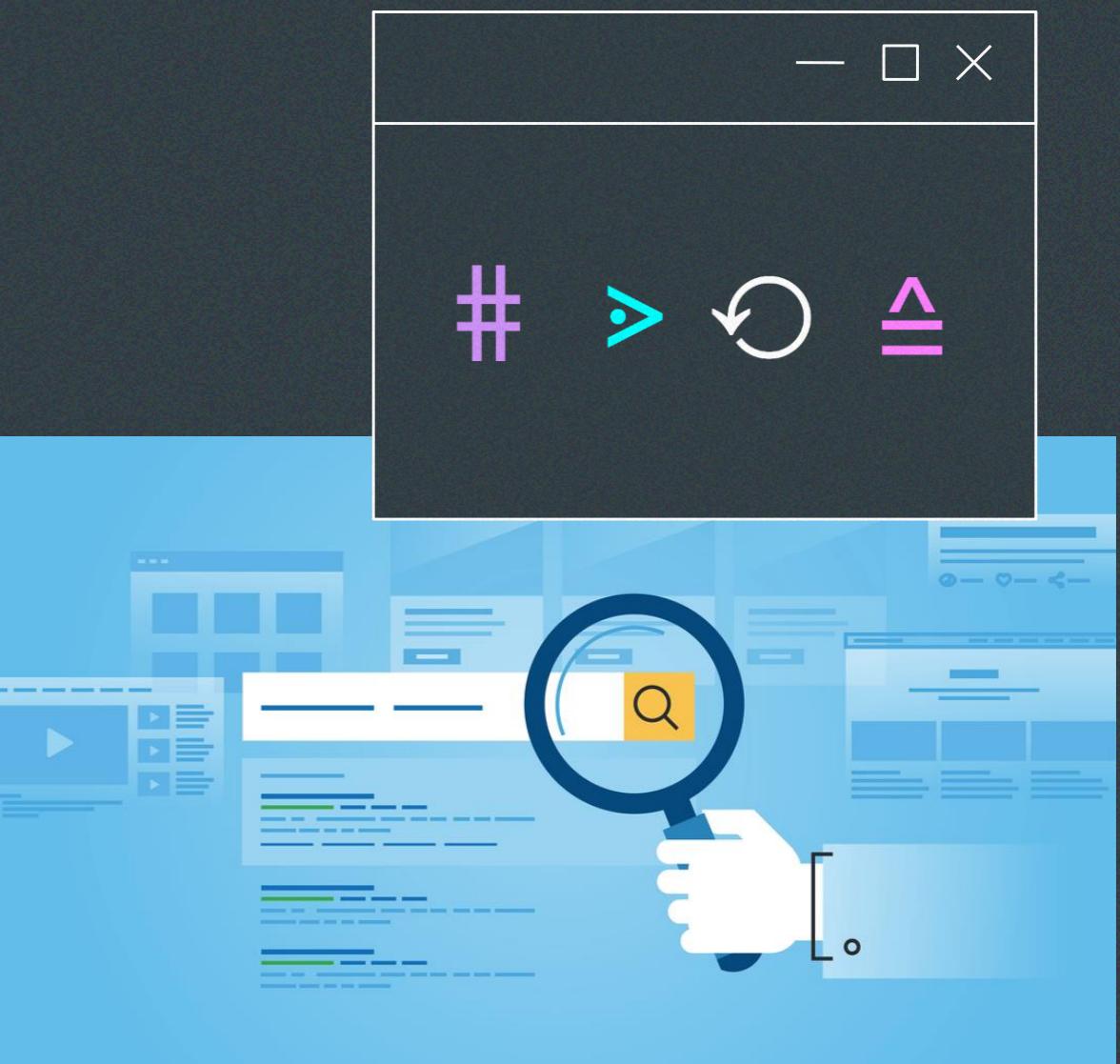
Implementarea PageRank în Motorul de Căutare **Google**

Inițial, clasarea paginilor web de către motorul de căutare Google a fost determinată de trei factori:

- Factori specifici paginii
- Textul de ancorare al linkurilor de intrare
- PageRank

Factorii specifici paginii sunt, pe lângă textul corpului, de exemplu **conținutul etichetei de titlu sau adresa URL** a documentului.

De la publicațiile Page și Brin, mai mulți factori s-au alăturat metodelor de clasare ale motorului de căutare Google.

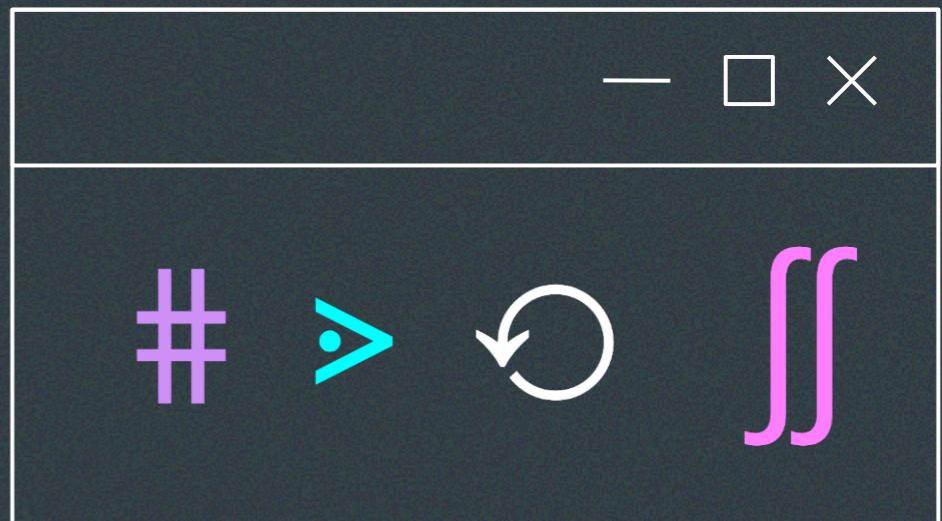


Implementarea (2)

Pentru a furniza rezultate de căutare, Google calculează un **scor IR** din factorii specifici paginii și textul de ancorare al linkurilor de intrare ale unei pagini, care este ponderat de poziția și accentuarea termenului de căutare în document.

Astfel se determină relevanța unui document pentru o interogare.

Scorul IR este apoi combinat cu PageRank ca indicator al importanței generale a paginii.



Pentru a combina scorul IR cu PageRank, cele două valori sunt **înmulțite**. Evident că acestea nu pot fi adăugate, deoarece în caz contrar paginile cu un PageRank foarte mare s-ar clasa în fruntea rezultatelor căutării chiar dacă pagina nu are legătură cu interogarea de căutare.

Efectul Linkurilor de Intrare

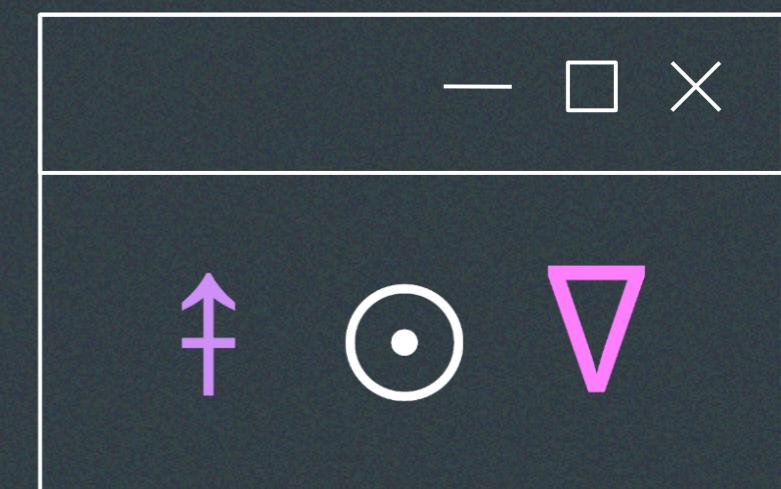
Fiecare link suplimentar de intrare pentru o pagină web crește întotdeauna PageRank-ul acelei pagini. Aruncând o privire la algoritmul PageRank, care este dat de

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

se poate presupune că un link de intrare suplimentar din pagina X crește PageRank-ul paginii A cu

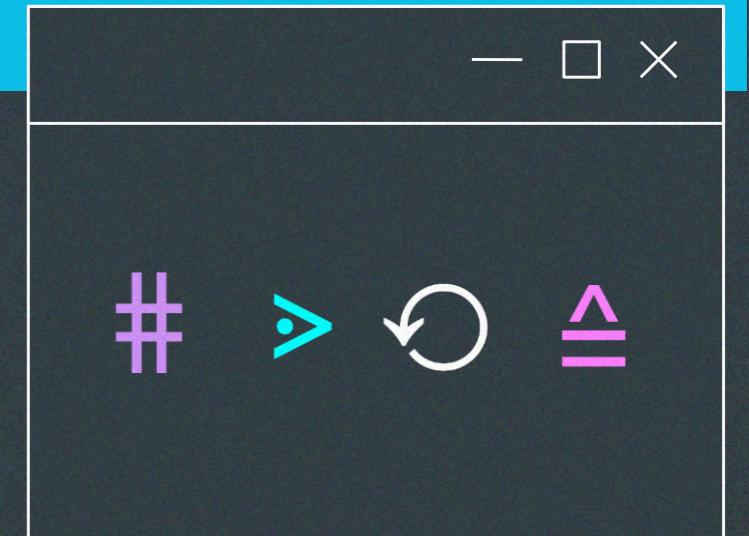
$$d \times PR(X) / C(X)$$

unde $PR(X)$ este PageRank al paginii X și $C(X)$ este numărul total de link-uri de ieșire ale acesteia.



Sfaturi pentru Creșterea Valorii PageRank a Site-ului vostru Web

- Adăugați pagini noi pe site-ul dvs. (cât de multe puteți)
- Schimbați link-uri cu site-uri web care au valoare PageRank ridicată
- Creșteți numărul de link-uri de intrare (Faceți publicitate site-ului dvs. pe alte site-uri)



Efectul Paginilor Adiționale

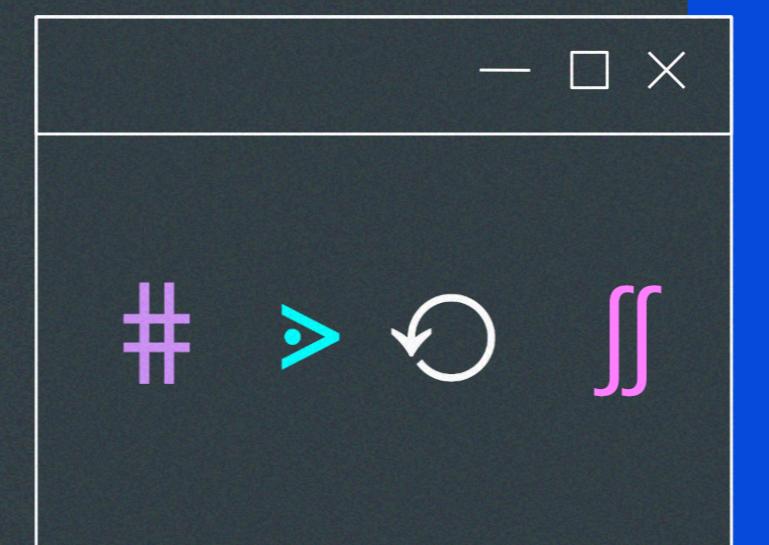
Sub-pages PageRank of the front page

1	1.000000
2	1.428673
3	1.857347
4	2.286020
5	2.714694
10	4.858060
20	9.144795
50	22.005003
100	43.438648
250	107.739838
500	214.907135
700	300.642426
1000	429.246613

După cum puteți vedea:

$\text{PageRank} \approx 1 + 0.428 * \text{nr_pagini}$

Deci dacă adăugați o pagină web pe site-ul vostru acesta va crește rangul paginii cu $\approx 0,428$. Desigur, trebuie să faceți aşa cum se arată în imagine



PageRank Example of 5 Pages After 2 Iterations



Pseudocod

```

1: function PageRank(G, m);
2:    $\pi = [1/m, 1/m, \dots, 1/m]$ ; //Distributia initiala de probabilitate
3:   eps =  $10^{-7}$ ;
4:   do
5:      $\pi' = \pi$ ;
6:      $\pi = \pi' * G$ ;
7:   while ( $\|\pi - \pi'\| \geq \text{eps}$ );
8:   return  $\pi$ ;
9: end function

```

$$G = \alpha \tilde{Q} + (1 - \alpha) \underbrace{\begin{bmatrix} 1/m & 1/m & \dots & 1/m \\ 1/m & 1/m & \dots & 1/m \\ \vdots & & & \\ 1/m & 1/m & \dots & 1/m \end{bmatrix}}_E.$$

	1	2	3	4	5	6
1	0	1/4	1/4	1/4	1/4	0
2	0	0	0	0	0	0
3	0	1/3	0	0	1/3	1/3
4	1/2	0	1/2	0	0	0
5	0	1/3	1/3	0	0	1/3
6	0	0	0	0	0	0

$$P_{ij} = 1/m$$

	1	2	3	4	5	6
1	0	1/4	1/4	1/4	1/4	0
2	1/6	1/6	1/6	1/6	1/6	1/6
3	0	1/3	0	0	1/3	1/3
4	1/2	0	1/2	0	0	0
5	0	1/3	1/3	0	0	1/3
6	1/6	1/6	1/6	1/6	1/6	1/6

PageRank in Python

```
import numpy as np
def pagerank(C, eps=0.0001, d=0.85):
    P = np.ones(len(C))
    while True:
        P_ = np.ones(len(A)) * (1 - d) + d * C.T.dot(P)
        delta = abs(P_ - P).sum()
        if delta <= eps:
            return P_
        P = P_
p=pagerank(C)
#result
#p=[1.16, 0.644, 1.19, 0.15]
```

Exemplu Concret

Imagine a scenario where there are 5 webpages A, B, C, D and E.

- Site A (outlinks to B, C, D)
- Site B (outlinks to A, C, D)
- Site C (outlinks to D)
- Site D (outlinks to C, E)
- Site E (outlinks to B, C, D)



Un script Python pentru vizualizarea rangului anumitor noduri prin algoritmul PageRank folosind un graf orientat ca și bază de date pentru noduri.

```
Number of iterations is: 5
Page rank of node  1 is :  0.7563090216610497
Page rank of node  2 is :  0.7570825988098917
Page rank of node  3 is :  1.021617613959075
Page rank of node  4 is :  0.9412927238508162
Page rank of node  5 is :  0.7735323180962704
Sum of all page ranks:  4.249834276377103
```

```

def win(matrix, m, o):
    k = 0
    for i in range(0, n):
        if(int(matrix[i][m]) == 1):
            k = k+1
    l = 0
    for i in range(0, n):
        if(int(matrix[o][i] == 1)):
            for j in range(0, n):
                if(matrix[j][i] == 1):
                    l = l+1
    return float(k/l)

```

```

def wout(matrix, m, o):
    k = 0
    for i in range(0, n):
        if(int(matrix[0][i]) == 1):
            k = k+1
    l = 0
    for i in range(0, n):
        if(int(matrix[o][i] == 1)):
            for j in range(0, n):
                if(matrix[i][j] == 1):
                    l = l+1
    return float(k/l)

```

```

def pagerank(matrix, o, n, p):
    a = 0
    for i in range(0, n):
        if(int(matrix[i][o]) == 1):
            k = 0
            for s in range(0, n):
                if(matrix[i][s] == 1):
                    k = k+1
            a = a+float((p[i]/k)*win(matrix, i, o)*wout(matrix, i, o))
    return a

n = 5
matrix = [[0, 1, 1, 1, 0], [1, 0, 1, 1, 0], [
    0, 0, 0, 1, 0], [0, 0, 1, 0, 1], [0, 1, 1, 1, 0]]
d = 0.25 # damping factor

o = 5
print("Number of iterations is:", o)

sum = 0
p = []

for i in range(0, n):
    p.append(1)
for k in range(0, o):
    for u in range(0, n):
        g = pagerank(matrix, u, n, p)
        p[u] = (1-d)+d*g
    for i in range(0, n):
        sum += p[i]
        print("Page rank of node ", i+1, "is : ", p[i])
print("Sum of all page ranks: ", sum)

```

BIBLIOGRAFIE

https://www.geeksforgeeks.org/weighted-pagerank-algorithm/?ref=ml_lbp

Emilia Petrișor, Probabilități și Statistică-Curs si Aplicații in Inginerie (ediția tipărită Editura Politehnica 2001)

<https://ro.wikipedia.org/wiki/PageRank>

<https://medium.com/@sarthakanand/page-rank-b7072c61dd85>

Mulțumim!

Echipa:

Ciobanu Daria

Doca Andrei

Georgescu Cătălin

Hondola Paul

Prof.Jivulescu Maria
Anastasia

÷ ≥ ↓↑