Capitolul 10

Data mining – cautare pe web

Căutarea pe web

- Puncte importante :
  1. *Rangul paginii*, pentru descoperirea celor mai "importante" pagini de web, utilizat de Google.
  2. *Indecşi şi autorităţi*, o evaluare mai detaliată a importanţei paginilor web utilizând o variantă a calculului de valori proprii utilizată pentru rangul paginii.

Rangul paginii

- Intuitiv rezolvăm problema definiţiei "importanţei" recursiv : o pagina este importantă dacă pagini inportante conţin legături către ea.
- Creăm o matrice stochastică a Internetului astfel :
  - Fiecare pagină *i* corespunde liniei *i* şi coloanei *i* a matricii.
  - Daca pagina *j* are *n* succesori (legături), atunci elementul *i, j* al matricii este *1/n* daca pagina *i* este unul dintre aceşti succesori ai paginii *j* şi 0 altfel.

Rangul paginii

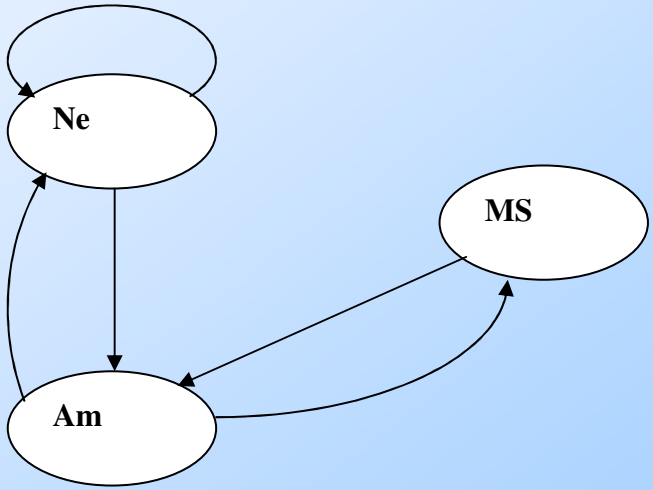
- Intuiţia care stă în spatele acestei matrici este :
- Să ne imaginăm că iniţial fiecare pagină are o unitate de importanţă.
- La fiecare pas fiecare pagină îşi împarte importanţa între succesorii săi şi primeşte noi fracţiuni de importanţă de la predecesorii săi.

Rangul paginii

- Eventual, importanţa fiecărei pagini atinge o limită care este componenta corespunzătoare ei din vectorul principal de valori proprii al matricii.
- Această importanţă este de asemenea probabilitatea ca un navigator pe web, pornind de la o pagină aleatoare şi urmând legături aleator alese din fiecare pagină, să ajungă la pagina în discuţie după o lungă serie de legături.

Exemplul 1

- In 1839 Internetul consta din doar trei pagini : Netscape, Amazon si Microsoft. Legăturile între aceste trei pagini erau ca în figura urmatoare:



Exemplul 1

- Fie [*n, m, a*] vectorul importanţei pentru cele trei pagini : Netscape, Microsoft respectiv Amazon.
- Atunci ecuaţia care descrie valorile asimptotice ale acestor trei variabile este :

$$\begin{bmatrix} n \\ m \\ a \end{bmatrix} = \begin{bmatrix} 1/2 & 0 & 1/2 \\ 0 & 0 & 1/2 \\ 1/2 & 1 & 0 \end{bmatrix} \begin{bmatrix} n \\ m \\ a \end{bmatrix}$$

Exemplul 1

- Prima coloană a matricii reflectă faptul că Netscape îşi divide importanţa între el însuşi şi Amazon. A doua coloană că Microsoft dă toată importanţa sa către Amazon.
- Putem rezolva ecuaţii ca aceasta începând cu aserţiunea că n = m = 1 şi aplicând repetat matricea la estimarea curentă a acestor valori.

Exemplul 1

- Primele patru iteraţii dau următoarele estimări :

n	=	1	1	5/4	9/8	5/4
m	=	1	1/2	3/4	1/2	11/16
a	=	1	3/2	1	11/8	17/16

- La limită, soluţia este $n = a = 6/5$; $m = 3/5$.

Observatii

- De notat că nu putem să obţinem niciodată valorile absolute ale lui *n*, *m* şi *a* ci ***doar raportul lor***, de vreme ce aserţiunea iniţială că fiecare a pornit de la 1 a fost arbitrară.
- Deoarece matricea este stochastică (suma pe fiecare coloană este 1), procesul de *relaxare* de mai sus converge către ***vectorul principal de valori proprii*** al matricii.

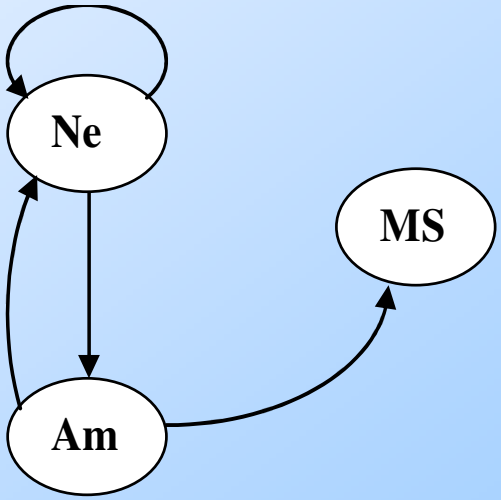
Probleme cu grafuri reale

- 2 tipuri de probleme:

1. *Dead end*: o pagină care nu are succesori nu are către cine să-şi trimită importanţa. Eventual, toată importanţa "se va scurge" din Internet
2. *Capcane* : un grup de una sau mai multe pagini care nu au legături către pagini din afara grupului vor acumula eventual toată importanţa din Internet.

Exemplul 2: Dead end

- Să presupunem că Microsoft încearcă să profite că este un monopol înlăturând toate legăturile din situl său. Noul Internet este ca in figura urmatoare:



Exemplul 2

- Ecuatia matriciala este in acest caz:

$$\begin{bmatrix} n \\ m \\ a \end{bmatrix} = \begin{bmatrix} 1/2 & 0 & 1/2 \\ 0 & 0 & 1/2 \\ 1/2 & 0 & 0 \end{bmatrix} * \begin{bmatrix} n \\ m \\ a \end{bmatrix}$$

- Se observa ca suma pe coloane nu mai este intotdeauna 1 (coloane cu suma nula)

Exemplul 2

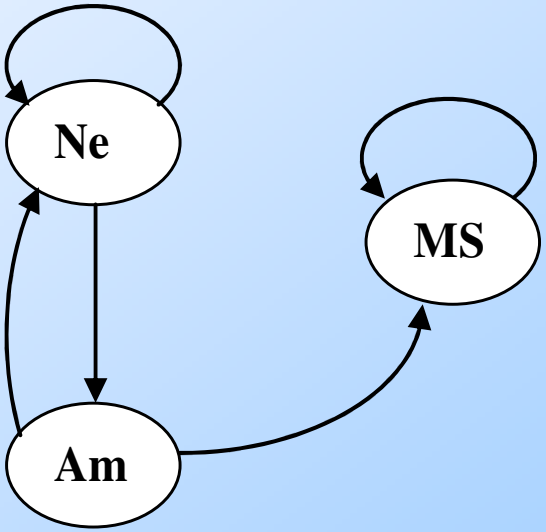
- Primii patru paşi ai soluţiei iterative sunt :

n	=	1	1	3/4	5/8	1/2
m	=	1	1/2	1/4	1/4	3/16
a	=	1	1/2	1/2	3/8	5/16

- In acest caz, fiecare dintre *n, m* şi *a* tinde catre 0; i.e. toată importanţa se scurge afară.

Exemplul 3

- Microsoft decide să nu folosească decăt legături către el însuşi de acum încolo. Acum, Microsoft a devenit o capcană. Noul Internet este in figura:



Exemplul 3

- Ecuatia matriciala este in acest caz:

$$\begin{bmatrix} n \\ m \\ a \end{bmatrix} = \begin{bmatrix} 1/2 & 0 & 1/2 \\ 0 & 1 & 1/2 \\ 1/2 & 0 & 0 \end{bmatrix} * \begin{bmatrix} n \\ m \\ a \end{bmatrix}$$

- Suma pe coloane este 1 dar apar valori de 1 pe diagonala principala a matricii.

Exemplul 3

- Primii pasi ai algoritmului produc valorile:

n	=	1	1	3/4	5/8	1/2
m	=	1	3/2	7/4	2	35/16
a	=	1	1/2	1/2	3/8	5/16

- Se observa acumularea pe linia m

Prevenire dead end şi capcane

- În loc de a aplica matricea direct, "taxăm" fiecare pagină cu o fracţiune din importanţa sa curentă şi distribuim importanţa taxată în mod egal tuturor paginilor.
- Dacă folosim o taxă de 20% ecuaţia din exemplul 3 devine cea de pe transparentul urmator.

- Ecuatia cu taxare:

$$\begin{bmatrix} n \\ m \\ a \end{bmatrix} = 0.8 \begin{bmatrix} 1/2 & 0 & 1/2 \\ 0 & 1 & 1/2 \\ 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} n \\ m \\ a \end{bmatrix} + \begin{bmatrix} 0.2 \\ 0.2 \\ 0.2 \end{bmatrix}$$

- Soluţia acestei ecuaţii este $n = 7/11$; $m = 21/11$; $a = 5/11$.
- De notat că suma celor trei valori nu este 3 dar obţinem o distribuţie mult mai rezonabilă a importanţei decăt în Exemplul 3.

Spam

- "Spamming" este în acest context încercarea multor situri web de a părea că sunt despre un subiect care atrage navigatorii fără ca într-adevăr să fie despre acel subiect.
- Google, ca şi alte motoare de căutare, încearcă să potrivească cuvintele din cererile de căutare cu cuvinte din pagini web.
- Cu toate acestea, Google, spre deosebire de alte motoare de căutare, tinde să creadă ceea ce spun alţii în textul legăturilor despre o pagină web facând mai greu pentru aceasta să pară ca fiind despre ceva ce nu este.

Spam

- Utilizarea rangului paginii pentru a măsura importanţa în locul unei măsuri mult mai naive ca "numărul de legături către acea pagină" protejează de asemenea impotriva spamului.
- Masura naivă poate fi inşelată de un spammer care creează 1000 de pagini care se referă între ele în timp ce rangul paginii recunoaşte că nici una dintre acestea nu au importanţă reală.

Indecşi si autorităţi

- Intuitiv, definim "index" şi "autoritate" într-un mod mutual recursiv: un index conţine legături către multe autorităţi iar o autoritate este referită de mulţi indecşi.
- Autorităţile pot fi pagini care oferă informaţii despre un subiect, e.g. pagina Quest despre proiectul IBM de data mining.
- Indecşii sunt pagini care nu furnizează informaţii ci spun unde se găsesc informaţii, e.g. pagina cursului CS345.

Indecşi si autorităţi

- Utilizează o formalizare matricială similară cu cea de la rangul paginii dar fără restricţia stochastică. Numărăm fiecare legătură ca 1, indiferent de câţi succesori sau predecesori are o pagină.
- Aplicarea repetată a matricii duce la divergenţă, dar putem introduce un factor de scalare pentru a ţine valorile calculate pentru gradul de "autoritate" sau de "indexare" pentru fiecare pagina între limite finite.

Indecşi si autorităţi

- Definim matricea *A* ale cărei linii şi coloane corespund paginilor web având elementul $A_{ij} = 1$ dacă pagina *i* referă pagina *j* si 0 altfel.
- De notat că A^T, transpusa lui *A*, arată ca matricea utilizată pentru calculul rangului paginilor dar A^T are 1 acolo unde matricea pentru rang are fracţii.

Indecşi si autorităţi

- Fie *a* si *h* doi vectori iar componenta lor *i* corespunde gradului de autoritate respectiv indexare a paginii *i*. Fie λ şi μ factorii de scalare corespunzători care vor fi calculaţi mai târziu. Atunci putem afirma că:
- $h = \lambda A a$. Adică gradul de indexare al fiecărei pagini este suma gradelor de autoritate ale tuturor paginilor referite, scalată cu λ.
- $a = \mu A^T h$. Adică gradul de autoritate al fiecărei pagini este suma gradelor de indexare ale tuturor paginilor care o referă, scalată cu μ.

Indecşi si autorităţi

- Din ecuatiile (1) şi (2) putem deduce folosind substituţia, două ecuaţii care leagă vectorii *a* si *h* doar de ei înşişi:

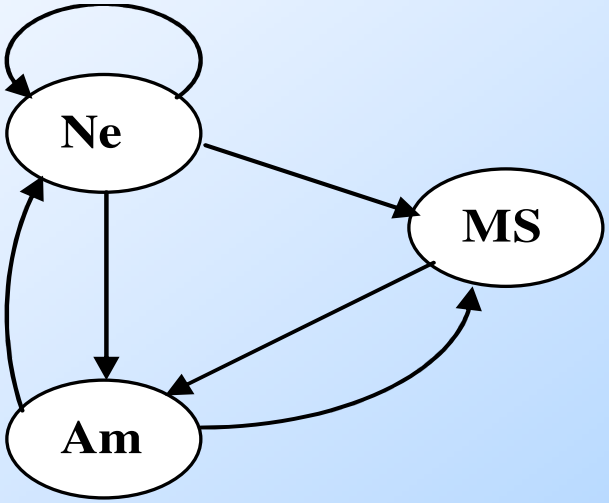
$$a = \lambda \mu A^T A a$$

$$h = \lambda \mu A A^T h$$

- Ca urmare, putem calcula h şi a prin relaxare, obţinând vectorul principal de valori proprii al matricilor AAT şi respectiv ATA

Exemplul 5

- Fie graful urmator:



Exemplul 5

- Matricile sunt:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad A^T = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad AA^T = \begin{bmatrix} 3 & 1 & 2 \\ 0 & 0 & 1 \\ 2 & 0 & 2 \end{bmatrix} \quad A^TA = \begin{bmatrix} 2 & 2 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

- Dacă utilizăm $\lambda = \mu = 1$ şi considerăm că vectorii $h = [h_n, h_m, h_a]$ şi $a = [a_n, a_m, a_a]$ sunt iniţial fiecare [1, 1, 1], primele trei iteraţii ale ecuaţiilor pentru a şi h sunt:

Exemplul 5

- Seccesiunea de valori pentru a este:

a_n	=	1	5	24	114
a_m	=	1	5	24	114
a_a	=	1	4	18	84

- Iar pentru h:

h_n	=	1	6	28	132
h_m	=	1	2	8	36
h_a	=	1	4	20	96

Exemplul 5

- Vectorul *a*, ***scalat corespunzător***, va converge către un vector în care
  - $a_n = a_m$ si
  - fiecare dintre aceste numere este mai mare ca a_a în raportul 1 + sqrt(3) / 2 sau aproximativ 1.36

Extragerea de cunostinţe din web

- Puncte importante:

1. *Numărarea dinamică a mulţimilor de articole*: Căutarea de mulţimi *interesante* de articole într-un spaţiu mult prea mare pentru a se putea lua în considerare fiecare pereche de articole.
2. *"Cărţi şi autori"*: Intrigantul experiment al lui Sergey Brin de extragere de date relaţionale din web.

Numarare dinamica

- Problema este de a găsi mulţimi de cuvinte care apar "neobişnuit de des" împreună pe web, e.g. "New" şi "York" sau {Ducesa, de, York}.
- "Neobişnuit de des" poate fi definit în diverse moduri pentru a încorpora ideea că numărul de documente web conţinând mulţimea de cuvinte este mult mai mare decât cel aşteptat în cazul în care cuvintele ar fi fost alese la întamplare, fiecare cuvânt cu probabilitatea sa de apariţie într-un document.

Numarare dinamica

- Un mod adecvat este *entropia per cuvânt din mulţime*. Formal, ***interesul*** unei mulţimi de cuvinte S este:

$$\frac{\log_2\left(\frac{prob(S)}{\prod_{w \text{ in } S} prob(\mathrm{w})}\right)}{|S|}$$

- De notat că împărţim la dimensiunea lui S pentru a evita "efectul Bonferroni", în care sunt atât de multe mulţimi de o dimensiune dată încât unele, din motive probabilistice, par a fi corelate.
- Exemplu: Daca *a*, *b* şi *c* (cuvinte) apar fiecare în 1% din toate documentele şi S={a, b, c} apar în 0.1% din documente, interesul lui S este (log2(0.001/(0.01 x 0.01 x 0.01))/3 = log2(1000)/3 adică aproximativ 3.3.

Numarare dinamica

- Problema tehnică: interesul nu este monoton sau "închis în jos" în modul de la produse cu larg suport.
- Asta înseamnă că putem avea o mulţime S cu o valoare mare a interesului şi totuşi unele sau chiar toate submulţimile sale stricte să nu fie interesante.
- Prin contrast, dacă S are suport larg, atunci toate submulţimile sale au cel puţin acelaşi suport.
- Observatie: Cu mai mult de 108 cuvinte diferite apărând pe web nu este posibil nici măcar să considerăm toate perechile de cuvinte.

DICE

- DICE (dynamic itemset counting engine) vizitează repetat paginile web într-un mod de tip "round-robin".
- De fiecare dată numără apariţiile anumitor mulţimi de cuvinte şi ale fiecărui cuvânt din aceste mulţimi.
- Numarul de mulţimi numărate este suficient de mic încât contorii lor încap în memoria centrală.

DICE

- Din când în când, să spunem la fiecare 5000 de pagini, DICE îşi reconsideră mulţimile pentru care numără. Înlătură acele mulţimi care au cel mai mic interes şi le înlocuieşte cu alte mulţimi.
- Alegerea noilor mulţimi se bazează pe proprietatea numita *heavy edge* care este o observaţie justificată experimental că acele cuvinte care apar în mulţimi cu interes ridicat au probabilitatea mai mare să apară în alte mulţimi cu interes ridicat.

DICE

- Astfel, când selectează noi mulţimi pentru a începe numărarea, DICE este direcţionat în favoarea cuvintelor care apar deja în mulţimi cu interes ridicat.
- Totuşi, el nu se bazează exclusiv pe aceste cuvinte altfel nu ar putea niciodată să găsească mulţimi cu interes ridicat compuse din multele cuvinte pe care nu le-a considerat niciodată.
- Unele (dar nu toate) din construcţiile pe care le utilizează DICE pentru crearea noilor mulţimi sunt:

DICE: noi multimi

1. Două cuvinte aleatoare. Aceasta este singura regulă independentă de aserţiunea muchiei grele şi ajuta noi cuvinte să ajungă în mulţime.
2. Un cuvânt dintr-una din mulţimile interesante şi un cuvânt aleator.
3. Două cuvinte din două mulţimi interesante diferite.

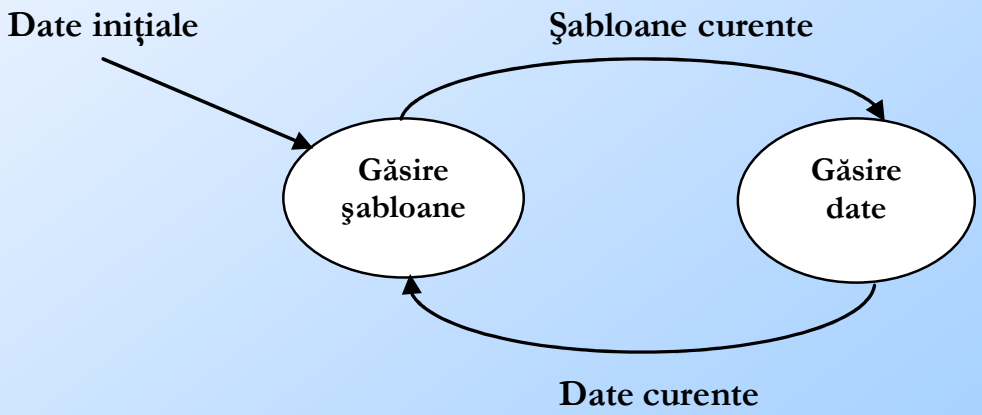
DICE: noi multimi

4. Reuniunea a două mulţimi interesante a căror intersecţie are dimensiunea 2 sau mai mult.
5. {a, b, c} dacă toate mulţimile {a, b}, {a, c} şi {b, c} sunt găsite ca fiind interesante.

- Bineînţeles, în general sunt mult prea multe opţiuni de a aplica cele de mai sus în toate modurile posibile astfel încât se utilizează o selecţie aleatoare a opţiunilor dând o anumită şansă fiecăreia dintre ele.

Carti si autori

- Ideea principală este de a căuta pe web fapte de un anumit tip, de genul celor care ar putea forma o relaţie de genul *Cărţi(titlu, autor)*. Procesarea este sugerată de figura urmatoare:



Cum lucreaza

1. Se porneşte de la un eşantion al tuplelor care se doresc găsite. În exemplul discutat în lucrarea lui Brin au fost folosite cinci exemple de titluri de cărţi şi autori ai acestora.
2. Pe baza exemplelor cunoscute, se caută pagini unde apar aceste date pe web. Dacă se găseşte un şablon care identifică un număr de tupluri cunoscute şi este suficient de specific încât e puţin probabil să identifice prea mult, atunci se acceptă acest şablon.

Cum lucreaza

3. Fiind dată o mulţime de şabloane acceptate, se caută date care satisfac aceste şabloane şi se adaugă la mulţimea datelor cunoscute.
4. Se repetă paşii (2) şi (3) de un număr de ori. În exemplul citat au fost utilizate patru ciclări care au dus la 15,000 de tuple; aprox. 95% au fost perechi adevărare titlu-autor.

Ce este un sablon

- Are 5 componente

1. *Ordinea*; i.e. daca titlul apare în text înaintea autorului sau vice-versa. Într-un caz general, în care tuplele au mai mult de 2 componente, ordinea va fi dată de permutarea componentelor.
2. *Prefixul adresei web (URL).*
3. *Prefixul* textului, care apare înaintea primului dintre titlu şi autor
4. *Mijlocul*: text care apare între cele două elemente de date.
5. *Sufixul* textului care urmează după al doilea dintre cele două elemente de date. Atât prefixul cât şi sufixul au fost limitate la 10 caractere.

Exemplu

- Un şablon posibil poate consta din următoarele:

1. **Ordinea**: titlul şi apoi autorul.
2. **Prefixul URL**: www.stanford.edu/class
3. **Prefixul**, mijlocul şi sufixul de forma următoare:

<LI><I>titlu</I> de autor<P>

Aici **prefixul** este <LI><I>, **mijlocul** este </I> de (inclusiv spaţiul după "de") şi **sufixul** este <P>. Titlul este orice apare între prefix şi mijloc; autorul este orice apare între mijloc şi sufix.

Tuning sablon

- Definim ***specificitatea*** unui şablon ca fiind produsul lungimilor prefixului, mijlocului, sufixului şi prefixului URL. În mare, specificitatea măsoară cât de posibil este să găsim date care corespund şablonului; cu cât specificitatea este mai mare, cu atât ne aşteptăm la mai puţine apariţii ale acestuia în date.
- Apoi şablonul trebuie să îndeplinească două condiţii pentru a fi acceptat:

1. Trebuie să fie cel puţin 2 elemente de date cunoscute care apar conform acelui şablon.
2. Produsul specificităţii şablonului cu numărul de apariţii de date conform acestuia trebuie să depăşească un anumit prag T (nespecificat).

Pasii executiei

1. Găsirea aparitiilor pornind de la datele cunoscute
2. Construcţia şabloanelor din aparitiile de date
3. Găsirea aparitiilor pornind de la şabloane

Aparitie

- O apariţie a unui tuplu este asociată cu un şablon după care acestea apar; i.e., acelasi titlu şi autor pot să apară după diferite şabloane. Astfel, o apariţie a datelor constă în:

1. Un anumit titlu şi autor.
2. Adresa Internet completa (URL) şi nu doar prefixul ca în cazul şablonului.
3. Ordinea, prefixul, mijlocul şi sufixul şablonului după care au apărut titlul şi autorul respectiv.

Constructia sabloanelor

1. Se grupează aparitiile de date după ordinea şi mijlocul lor. De exemplu, un grup din acest "group-by" poate corespunde ordinii "titlu-apoi-autor" şi mijlocului "</I> de ".
2. Pentru fiecare grup se găseşte cel mai lung prefix, sufix şi prefix URL comun.
3. Dacă testul de specificitate pentru acest şablon este îndeplinit, se acceptă şablonul.

Constructia sabloanelor

4. Dacă testul de specificitate *nu* este îndeplinit, se încearcă spargerea grupului în doua prin extinderea lungimii prefixului URL cu un caracter şi apoi se repetă pasul (2). Dacă este imposibil să spargem grupul (pentru că există doar un URL) atunci am eşuat în a produce un nou şablon din acel grup.

- **Exemplu:** Să presupunem că grupul conţine trei URL-uri:
  - www.stanford.edu/class/cs345/index.html
  - www.stanford.edu/class/cs145/index.html
  - www.stanford.edu/class/cs340/readings.html

Constructia sabloanelor

- Prefixul comun este www.stanford.edu/class/cs.
- Dacă trebuie să spargem grupul, atunci următorul caracter, 3 sau 1, sparge grupul în două, cu acele apariţii ale datelor din prima pagină (pot fi multe astfel de apariţii) mergând într-un prim grup şi apariţiile din celelalte două pagini în celălalt:
  - www.stanford.edu/class/cs345/index.html
  - www.stanford.edu/class/cs340/readings.html

  Si
  - www.stanford.edu/class/cs145/index.html

Gasire aparitii din sabloane

1. Se găsesc toate URL-urile care se potrivesc cu prefixul URL al cel puţin unui şablon.
2. Pentru fiecare astfel de pagină se parcurge textul folosind o expresie regulată construită din prefixul, mijlocul şi sufixul şablonului.
3. Se extrage din fiecare potrivire titlul şi autorul, dupa ordinea specificată în şablon.

Bibliografie

- J.D.Ullman - CS345 --- Lecture Notes:

  PageRank, Hubs-and-Authorities , Web Mining

  http://infolab.stanford.edu/~ullman/cs345-notes.html

Sfârşitul capitolului 10