dPersp Uge 41 - Øvelser Internetalgoritmer

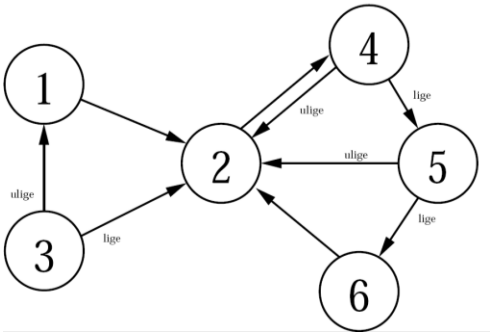
(Øvelserne 4 og 6 er afleveringsopgaver)

Øvelse 1

a) Hver gruppe får en terning af instruktoren.

Udfør 100 skridt af nedenstående RandomWalk på grafen, som også findes på slidsene fra mandagens forelæsning. For hvert skridt noter i hvilken af de 6 knuder I står. Når I har foretaget 100 skridt, tæl da op hvor mange gange I har været i hver af de 6 knuder. Beregn den procentvise fordeling blandt de seks knuder. Sammenlign med de beregnede sandsynligheder.

Metode RandomSurfer
Start på knude 1
Gentag mange gange:
    Kast en terning:
    Hvis den viser 1-5:
        Vælg en tilfældig pil ud fra knuden
        ved at kaste en terning hvis 2 udkanter
    Hvis den viser 6:
        Kast terningen igen og spring hen til den knude
        som terningen viser



Knude	1	2	3	4	5	6
Antal gange						
% fordeling						
(% fra slides)	0.039	0.353	0.028	0.322	0.162	0.095

b) Læg tallene for alle grupperne på holdet sammen og sammenlign igen med de beregnede sandsynligheder.

Knude	1	2	3	4	5	6
Gruppe 1						
Gruppe 2						
Gruppe 3						
...						
Totalt						
% fordeling						
(% fra slides)	0.039	0.353	0.028	0.322	0.162	0.095

Øvelse 2

Regnearket hvormed PageRank værdierne for eksemplet i øvelse 1 er beregnet med findes her:

www.cs.au.dk/~gerth/dPersp/pagerank.xls

I denne opgave skal I downloade regnearket og modificere det. Cellerne som det er nødvendigt at rette i for at løse de følgende øvelser er markeret med gult. Bemærk at der er to worksheets: Et med grafen og et med PageRank beregningerne.

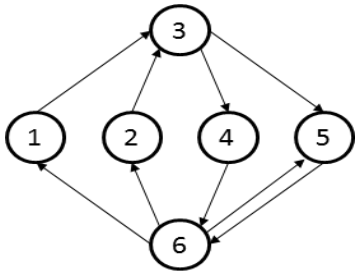
a) Prøv at ændre sandsynligheden for at foretage et tilfældigt spring (skal være i intervallet [0,1]). Hvor mange skridt skal man gentage før de første 3 decimaler i sandsynlighedsfordelingen ikke ændrer sig? Gentag beregningerne for alle sandsynlighederne fra 0,0 til 1,0 med skring på 0,1.

sandsynlighed	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
# skridt											

b) Gentag ovenstående beregning for grafen med nedenstående incidensmatrice (hvilket er en 0-1-matrice hvor indgang (i,j) indeholder 1 hvis og kun hvis der er en kant (*i,j*) i grafen). Læg specielt mærke til beregningen for sandsynligheden 0,0 for at foretage et tilfældigt spring – dvs. man følger kun kanterne tilfældigt uden nogensinde at foretage et spring til en tilfældig knude. Prøv at forklare jeres observationer.

www.cs.au.dk/~gerth/dPersp/pagerank2.xls

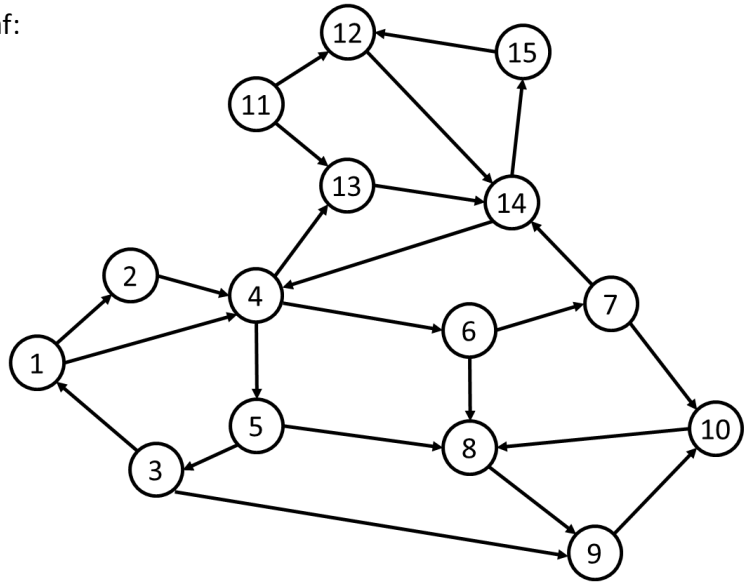
(i,j)	1	2	3	4	5	6
1	0	0	1	0	0	0
2	0	0	1	0	0	0
3	0	0	0	1	1	0
4	0	0	0	0	0	1
5	0	0	0	0	0	1
6	1	1	0	0	1	0



sandsynlighed	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
# skridt											

Øvelse 3

Betragt følgende graf:



Regnearket til beregning af PageRank værdierne findes her:

www.cs.au.dk/~gerth/dPersp/pagerank3.xls

a) Hvilket link skal tilføjes for at maksimere PageRank værdien for knuden 1?

b) Hvad er det mindste antal links (og hvilke) der skal tilføjes grafen for knuden 1 får den højeste PageRank værdi?

Knude	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
PageRank															

Øvelse 4 (Afleveringsopgave 1)

Detaljerne i Google's rangering af søgeresultater er en af deres dybeste hemmeligheder. PageRank er kun en af de mulige parametre.

Prøv at overveje hvordan I ville rangere en mængde af søgeresultater, og beskriv hvordan I ville forsøge at rangere outputtet så relevante websider bliver listet først. Mulige emner I kan komme ind på (men behøves ikke) er f.eks. hvordan håndteres personlig rangering, kunstige for høje rangeringer (modvirk SEO), geografisk information, tvetydige søgninger (f.eks. "frø"), …

I må meget gerne søge information på nettet til at besvare opgaven.

Øvelse 5

a) Beskriv **map** og **reduce** funktioner der kan bruges til at transformere en liste af n værdier

$$[\ x_1, x_2, \ldots, x_n\ ]$$

til en liste med kun én værdi

$$[\ k\ ]$$

hvor k er antal **forskellige** x_i i listen, dvs. vi beregner antal forskellige værdier i input.

b) Beskriv **map** og **reduce** funktioner der kan bruges til at transformere en liste af n par

$$[\ (x_1,c), (x_2,0), \ldots, (x_n,0)\ ]$$

til listen

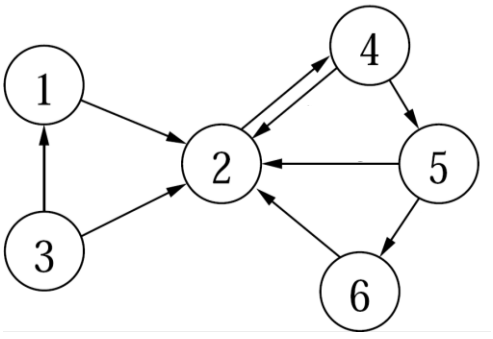
$$[\ (x_1,c), (x_2,c), \ldots, (x_n,c)\ ]$$

dvs. tilknytter til hvert x_i værdien c. Vi antager $c > 0$.

Øvelse 6 (Afleveringsopgave 2)

I denne opgave vil vi se på hvordan man kan anvende MapReduce interfacet til at beregne PageRank værdierne for en webgraf. Vi antager at inputtet er givet ved en liste af links (i,j), som angiver at side i henviser til side j, og hvor både i og j antages at være heltal. Vi antager at hver side indeholder mindst ét link. For nedenstående graf har vi f.eks. følgende liste som input:

[(1,2), (5,6), (2,4), (3,1), (4,2), (4,5), (6,2), (5,2), (3,2)]



Vi ønsker at beregne sandsynlighedsfordelingen for at være på siderne efter s = 50 skridt af RandomSurfer algoritmen, dvs. vi ønsker at beregne listen

$$[\ (i_1, p_{i_1}^{(s)}), (i_2, p_{i_2}^{(s)}), (i_3, p_{i_3}^{(s)}), \ldots, (i_n, p_{i_n}^{(s)})\ ]$$

hvor $i_1, i_2, \ldots, i_n$ er de n forskellige sider der indgår i linksene. Sandsynlighederne beregnes ud fra følgende formel:

$$p_1^{(0)} = 1.0 \qquad p_2^{(0)} = \cdots = p_n^{(0)} = 0.0 \qquad p_i^{(s)} = 0.85 \cdot \sum_{j\,:\,j \to i} \frac{p_j^{(s-1)}}{\text{udgrad}(j)} + 0.15 \cdot \frac{1}{n} \qquad (*)$$

For ovenstående eksempel graf bliver dette

$$[\ (1, 0.03563),\ (2, 0.35462),\ (3, 0.02500),\ (4, 0.32643),\ (5, 0.16373),\ (6, 0.09459)\ ]$$

(værdierne er beregnet v.h.a. regnearket fra PageRank øvelse 1 ved at sætte sandsynligheden til 15%).

Vi kan beregne den ønskede liste ved at foretage nedenstående transformationer på vores input liste, hvor ①, ②, ③, og ⑤ udføres præcis én gang, og ④ udføres s (=50) gange. ① udvider hvert link (i,j) med sandsynligheden for at stå på side i i starten af RandomSurfer processen, ② udvider yderligere hvert link (i,j) med information om udgraden af i i webgrafen, og endeligt udvider ③ hvert link med information om det totale antal sider n repræsenteret i input. ④ beregner for et link (i,j) den nye sandsynlighed for at stå på side i hvis vi laver yderligere ét skridt i RandomSurfer algoritmen. Dette kan beregnes ved formelen (*) ovenfor. Endeligt reducerer ⑤ input til et par for hver knude.

$$[\ (i_1, j_1), (i_2, j_2), \ldots\ ]$$

$$\xrightarrow{①} [\ (i_1, j_1, p_{i_1}^{(0)}), (i_2, j_2, p_{i_2}^{(0)}), \ldots\ ]$$

$$\xrightarrow{②} [\ (i_1, j_1, p_{i_1}^{(0)}, \text{udgrad}(i_1)), (i_2, j_2, p_{i_2}^{(0)}, \text{udgrad}(i_2)), \ldots\ ]$$

$$\xrightarrow{③} [\ (i_1, j_1, p_{i_1}^{(0)}, \text{udgrad}(i_1), n), (i_2, j_2, p_{i_2}^{(0)}, \text{udgrad}(i_2), n), \ldots\ ]$$

s ×

$$\xrightarrow{④} [\ (i_1, j_1, p_{i_1}^{(1)}, \text{udgrad}(i_1), n), (i_2, j_2, p_{i_2}^{(1)}, \text{udgrad}(i_2), n), \ldots\ ]$$

$$\xrightarrow{④} [\ (i_1, j_1, p_{i_1}^{(2)}, \text{udgrad}(i_1), n), (i_2, j_2, p_{i_2}^{(2)}, \text{udgrad}(i_2), n), \ldots\ ]$$

$$\ldots$$

$$\xrightarrow{④} [\ (i_1, j_1, p_{i_1}^{(s)}, \text{udgrad}(i_1), n), (i_2, j_2, p_{i_2}^{(s)}, \text{udgrad}(i_2), n), \ldots\ ]$$

$$\xrightarrow{⑤} [\ (i_1, p_{i_1}^{(s)}), (i_2, p_{i_2}^{(s)}), \ldots\ ]$$

a) Beskriv hvordan **mindst to** af de fem transformationer ①, ②, ③, ④ og ⑤ kan implementeres v.h.a. MapReduce interfacet for passende valg af **map** og **reduce** funktioner.