Trabajo Práctico

Recuperación de Información Web y Motores de Búsqueda

Bibliografía: [SE] Capítulos 3 y 4, [MAN] Capítulo 19, 20, 21
RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1
Papers: Brin, S. & Page, L. The Anatomy of a Large-Scale HyperTextual Web Search Engine. 1998.
Kleinberg, J. Authoritative sources in a hyperlinked environment. 1998.

1) ¿Qué es el standard de exclusión de robots? ¿Cómo funciona? Brinde un ejemplo del archivo robot.txt y explique su contenido.

2) ¿Cuáles son los códigos HTTP? ¿En qué caso se usa cada uno?

3) ¿Cómo se determinan las páginas dinámicas y las estáticas? ¿Qué diferencias hay? ¿Brinde un ejemplo (escriba un fragmento de código que funcione en su servidor HTTP) de página dinámica y explique cómo se pueden pasar los parámetros?

4) ¿Qué problemas hay con el servicio DNS en el crawling? ¿Cómo se resuelven?

5) ¿Qué expecifican los META tags NOINDEX y NOFOLLOW? ¿Dónde se utilizan?

6) Escriba un pequeño programa Perl que reciba como parámetro una URL y descarge la página HTML correspondiente. Luego, extraiga de la misma los enlaces y los muestre por consola (puede utilizar el módulo LWP::Simple).

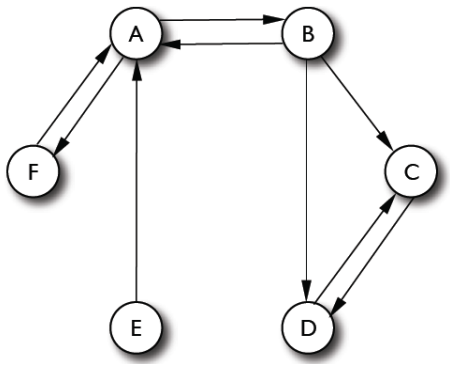
7) Modifique su programa anterior para implementar un Crawler básico de acuerdo al algoritmo siguiente. Realice una pequeña recolección (no más de 50 páginas) y con la información de enlaces arme el grafo correspondiente. Grafique utilizando la librería GraphViz[1].

1 http://www.graphviz.org/

```
1: push(todo_list,initial_set_of_urls)
2: while todo_list[0] ≠ ∅ do
3:    page ← fetch_page(todo_list[0])
4:    if page downloaded then
5:       links ← parse(page)
6:       for all l in links do
7:          if l in done_list then
8:             push(todo_list[0].outlinks,done_list[l].id)
9:          else if l in todo_list then
10:            push(todo_list[0].outlinks,todo_list[l].id)
11:         else if l pass our filter then
12:            push(todo_list,l)
13:            todo_list[l].id = no. of url's
14:            push(todo_list[0].outlinks,todo_list[l].id)
15:         end if
16:      end for
17:   end if
18: end while
```

9) Suponga que se han recuperado un conjunto de páginas de un pequeño repositorio de sólo seis utilizando el modelo de espacio vectorial. Dado un query Q se produjo la siguiente salida con los valores de sim(Q, Di). Las páginas se encuentran vinculadas de acuerdo al grafo de la derecha.

Pos	Doc	Sim
1	E	4.9734
2	C	4.8173
3	A	2.5617
4	B	2.0110
5	D	0.8937
6	F	0.0000



a) Escriba la matriz de adjacencia que representa la red (es decir, el paso inicial)

b) Calcule los valores de PageRank de las páginas utilizando como factor de damp 0.5, 1.0 y 1.5. Pruebe iterando 2, 5 y 10 veces. Para el cálculo de PageRank puede utilizar el módulo Perl Algorithm-PageRank-0.08[2]. Este módulo requiere PDL (Perl Data Language)[3]

Un ejemplo se uso del módulo:

```
use Algorithm::PageRank;
$pr = new Algorithm::PageRank;

$pr->graph([
        0 => 1,
        0 => 2,
        1 => 0,
        2 => 1,
        ]
        );

$pr->iterate(10);
print $pr->result();
```

c) Use los valores de PageRank para re-rankear la salida de la búsqueda. Puede combinar el score de Sim con el PageRank multiplicandolos. ¿Se altera el ranking? ¿En qué caso?

10) Considere la siguiente estructura de páginas enlazadas

Página A tiene un outlink a B y C
Página B tiene un outlink a C
Página C tiene un outlink a A

Ejecute PageRank (con a = 0.15) sobre este subgrafo. Haga 3 iteraciones y muestre en cada una los scores (con y sin normalización)

11) Considere la siguiente estructura de páginas enlazadas

Página A tiene un outlink a C, D y E
Página B tiene un outlink a D y E

Ejecute HITS sobre este subgrafo. Haga 3 iteraciones y muestre en cada una los scores (con y sin normalización)

2 http://search.cpan.org/~xern/Algorithm-PageRank-0.08
3 http://pdl.perl.org/download/ (En Debian se puede instalar mediante **apt-get install pdl**)