Estructuras de Datos

Universidad Nacional de Tres de Febrero

Trabajo Práctico 2

Enunciado

En este trabajo práctico vamos a desarrollar un buscador, para lo cual será necesario recorrer la web e indexar los documentos. Los módulos que tendrá nuestra aplicación serán los siguientes:

- Módulo Crawler
- Módulo Indice Invertido
- Módulo Buscador

Módulo Crawler

Este módulo será el encargado de recorrer la web. El archivo de configuración tendrá una sección especial `[CRAWLER]` los parámetros de configuración son:

`URLs:` lista de URLs que determinan la frontera del Crawler, separadas por punto y coma ";"

`Log:` Archivo donde se loggearan todas las páginas visitadas

`Tmin:` Tiempo mínimo que deberá respetar entre dos peticiones seguidas al mismo servidor

El crawler deberá tomar cada una las URLs de su frontera, visitar la página, extraer los enlaces a otras páginas y si estas páginas se encuentran dentro de su frontera deberá visitarlas, si uno de estos enlaces apunta a una página fuera de su frontera deberá ignorar el enlace. El crawler deberá ser **Robusto** es decir no debe caer en bucles infinitos (por ejemplo si dos páginas o más páginas se referencian mutuamente) o si un servidor de aplicaciones genera páginas dinámicamente creando una trampa. La ejecución de este módulo se deberá poder interrumpir en cualquier momento con CTRL-C, y se deberá poder reiniciar en cualquier momento. Cuando se reinicia no debe empezar todo de nuevo, sino que deberá revisar el archivo de Salida para no visitar nuevamente las páginas que haya visitado anteriormente. También deberá ser **Cortez** es decir debe mantener solo una conexión abierta a un servidor dado, esperar un tiempo mínimo establecido por el parámetro Tmin entre dos peticiones seguidas al mismo servidor, y en cada ejecución del crawler visitar cada página solo una vez. Por ejemplo si varias páginas de un sitio contienen un enlace a una página de contacto, sólo podrá visitar la página de contacto una única vez. También deberá registrar en una bitácora todas las acciones que realice. Cuando el módulo crawler comienza a ejecutarse deberá agregar la siguiente línea al archivo de log:

Crawler iniciado [dd/mm/aaaa hh:mm:ss] indicando la fecha y hora.

Cuando el crawler termina de ejecutarse o se interrumpe su ejecución deberá registrar:

Crawler finalizado [dd/mm/aaaa hh:mm:ss]

y cada vez que visita una página (realiza una petición a un servidor) deberá registrar:

url_visitada [dd/mm/aaaa/ hh:mm:ss]

Si dentro de una URL inicial que contenía en su frontera no hay más enlaces por visitar deberá registrar la siguiente línea:

url_inicial completa [dd/mm/aaaa/ hh:mm:ss]

El archivo de log así definido debe permitir reconstruir el orden y el momento en que visitó todas y cada una de las páginas.

En todos los casos el formato de fecha y hora deberá corresponder con la zona horaria de Argentina GMT-3

El módulo crawler deberá extraer el texto de cada una de las páginas que visita y pasarlo al módulo de índice invertido para que lo registre. Para realizar esta operación se deberá utilizar expresiones regulares.

A continuación se incluye una clase denominada `LinkParser` que permite visitar una URL y devuelve la lista de URLs encontradas en esa dirección y el HTML crudo.

In [ ]:
```
from html.parser import HTMLParser
from urllib.request import urlopen
from urllib import parse

class LinkParser(HTMLParser):
    def handle_starttag(self, tag, attrs):
        if tag == 'a':
            for (key, value) in attrs:
                if key == 'href':
                    newURL=parse.urljoin(self.baseURL, value)
                    self.links.append(newURL)

    def fetch_page(self, url):
        self.links=[]
        self.baseURL = url
        response = urlopen(url)
        contentType=response.getheader('Content-type')
        if 'text/html' in contentType:
            encoding=response.headers.get_param('charset')
            data=response.read()
            htmlString=data.decode(encoding)
            self.feed(htmlString)
            return htmlString, self.links
        else:
            return "",[]

if __name__ == '__main__':
    parser=LinkParser()
    resultados=parser.fetch_page("http://www.untref.edu.ar")
    print("página visitada: \n\n", resultados[0])
    print("\n\nEnlaces encontrados: \n\n", resultados[1])
```

Módulo İndice Invertido

Este módulo será el responsable de mantener un índice invertido con todas las palabras encontradas en todas las páginas que visite el Crawler.

La estructura de base deberá ser un ***árbol B con todos los datos en las hojas*** y deberá implementar las técnicas de compresión que veremos en clase.

El índice deberá mantenerse en memoria y será utlizado por el módulo buscador para realizar consultas.

El índice invertido se deberá construir ignorando STOP WORDS, palabras de longitud menor a `min_long` , variable definida en el archivo de configuración en la sección `[INVERTED_INDEX]`

Módulo Buscador

Este módulo será la interfaz con el usuario, el cual podrá ingresar palabras a buscar y el sistema deberá devolver las páginas donde se encuentra esa palabra. Deberá soportar los siguientes comodines en la consulta:

- *

Las consulta consistirá en palabras separadas por espacios y el sistema deberá responder con las páginas que contengan todas las palabras que cumplen con la condición. Ejemplo:

---

*tref computación

Debe devolver las páginas que contengan palabras que terminan en tref y computación

---

Archivo de configuración

La aplicación deberá ser capaz de leer un archivo de configuración con el siguiente formato (se recomienda utilizar el módulo configparser). Deberá contener al menos las secciones y variables indicadas anteriormente y se le podrán agregar más variables y secciones Ejemplo, dado el siguiente archivo denominado "config.ini"

```
[CRAWLER]
URLs = http:\\www.untref.edu.ar ; http:\\www.uba.ar
Log  = bitacora.log
Tmin = 5

[INVERTED_INDEX]
min_long = 3
```

In [ ]:
```
import configparser

configuracion=configparser.ConfigParser()

configuracion.read("config.ini")

print(configuracion["CRAWLER"]["URLs"])
print(configuracion["CRAWLER"]["Log"])
print(configuracion["CRAWLER"]["Tmin"])

print(configuracion["INVERTED_INDEX"]["min_long"])
```