Progetto per il corso di Laboratorio di Programmazione

MRQG: Un semplice Motore di Ricerca basato su Google

Introduzione al Progetto

Nicola Galesi
Stefano Guerrini
A.A. 2004/2005 – II Semestre

1 Preliminari

Lo scopo del progetto è quello di disegnare in più moduli un programma - MRQG - scritto su più files e in linguaggio C che implementi un motore di ricerca che, con le dovute semplificazioni, realizzi alcune delle caratteristiche implementative di Google.

Data una *query* - un testo composto da una o più parole - ed un insieme di pagine web, il programma che si realizzerà dovrà individuare tutte le pagine web dell'insieme iniziale dove una o più parole del testo compaiono. Questa sequenza di pagine web deve essere restituita ordinata secondo una misura chiamata *rank*. Il rank di una pagina web verrà calcolato usando differenti informazioni. Da un lato si valuterà la rilevanza di una pagina rispetto ad una query, considerando il tipo, lo stile, la posizione e il numero di occorrenze delle parole cercate nella pagina. Dall'altro ad ogni pagina web il programma dovrà associare una quantità, introdotta in Google e chiamata *pagerank*, che da una misura della "visibiltà" della pagina nel web. Il rank verrà infine calcolato combinando il pagerank con la valore di rilevanza della pagina rispetto alla query.

Di seguito si fornisce un'introduzione iniziale al progetto, e al tipo di semplificazioni e restrizioni considerate rispetto a Google. Una prima ed importante semplificazione riguarda il tipo di pagine web a cui applicare MRQG e la loro ubicazione.

Formato semplificato delle pagine web

Le pagine web in cui MRQG cercherà saranno solo ed unicamente ottenute mediante files di testo con estensione *.html* scritti in un linguaggio html **molto semplificato** che si descrive di seguito. Ogni pagina sarà composta da un **titolo**, che da il titolo alla pagina, e da un **corpo** che rappresenta il contenuto della pagina. Si tenga presente che tali pagine web semplificate non saranno visualizzabili usando un browser.

[nomefile]:= nome completo di file;
[separatore] := una sequenza di caratteri dell' insieme {'b','.','.'....'/cr'};
[parola] := sequenza alfanumerica;

[testo semplice] := sequenza di [parola] separate da un [separatore];
[testo corsivo]:= ¡s¿ [testo semplice] ¡/s¿;
[testo neretto] := ¡b¿ [testo semplice] ¡/b¿;
[testo normale] := Una sequenza formata da [testo semplice] e/o [testo corsivo] e/o [testo neretto]
[link] := ¡ a href = "nome file"¿ [testo normale] ¡/a¿
[testo] := Una sequenza formata da [testo normale] e/o [link]
[pagina web] := ¡title¿ [testo semplice] ¡/title¿ ¡body¿ [testo] ¡/body¿

Un esempio di file di testo per una pagina web semplificata è il seguente

```
<title>
Impresa Fratelli Rossi
</title>

<body>
L'impresa dei F. Rossi è sita in via Puccini 36 in <b> Roma </b>,
<a href = ''Descrizione.html''> Cliccare qui </a>
per una descrizione dettagliata della società.

Contattare il telefono <s> 99999999 </s> per
ulteriori informazioni.
</body>
```

Faremo l'ulteriore ipotesi che le pagina web a cui applicare il nostro motore di ricerca siano già **tutte** salvate in un'unica directory di lavoro. Il programma non dovrà per tanto cercare le pagine web in rete ne occuparsi di un loro eventuale aggiornamento. I docenti forniranno un numero di pagine web scritte nel precedente linguaggio per testare il programma.

Il grafo dei link e il pagerank

Una pagina web puo avere un link ad una o più differenti pagine web. Nella pagina dell'esempio precedente, il testo "Cliccare qui" è un link che permette di accedere alla pagina web corrispondente al file "Descrizione.html". Seguendo la filosofia di Google useremo la struttura dei link tra le differenti pagine per assegnare ad ogni pagina web una quantità - chiamata *pagerank* - che misuri l'importanza di tale pagina nel web.

Il pagerank di una pagina si può pensare in differenti modi. Si può considerare come la probabilità che un surfer che clicca sui links aleatoriamente senza tornare mai indietro, visiti quella pagina. Intuitivamente quindi una pagina web avrà un pagerank alto se ci sono molte altre pagine che puntano a quella pagina e che hanno a loro volta un pagerank alto. In definitiva pagerank cerca di implementare la filosofia per cui la pagine web più meritevoli di essere considerate sono quelle più visitate o quelle puntate da pagine web a loro volta molto visitate.

Per calcolare il pagerank immaginiremo la struttura dei link tra le pagine web codificata in un grafo diretto. Ogni nodo del grafo corrisponderà una pagina web e tra due nodi A e B avremo un arco orientato, che denoteremo con

(A, B), se nella pagina corrispondente al nodo A vi almeno un link alla pagina corrispondente al nodo B. Dato un nodo A nel grafo chiamiamo F_A l'insieme di nodi B a cui il nodo A è linkato, vale a dire tali che (A, B) è un arco orientato nel grafo dei link. Similmente, dato un nodo A denoteremo con B_A. l'insieme dei nodi che hanno un link ad A, cioè tali che (B, A) è un arco nel grafo dei link.

Il *Pagerank* $PR(A)$ di una pagina/nodo A verrà calcolato usando i precedenti insiemi. Un algoritmo ricorsivo per calcolarlo si deduce facilmente dalla seguente formula codifica le carettistiche intuitive del pagerank descritte sopra:

$$PR(A) = \sum_{B \in B_A} \frac{PR(B)}{|F_A|}$$

Nel modulo che si occuperà di calcolare il pagerank, vedremo alcuni problemi che presenta questo algoritmo e come risolverli abbastanza facilmente.

Dizionario degli ipertesti

Nella prima parte del progetto ci occuperemo di analizzare tutte le pagine web presenti nella directory di lavoro e di costruire un dizionario degli ipertesti che verrà implementato usando una tavols hash. Per ogni pagina web tale dizionario deve contenere tutte le differenti parole che occorrono nel testo della pagina e per ognuna di tali parole una lista, la *hit list*, con informazioni specifiche - come posizione nel testo, formato ecc. - delle differenti occorrenze della stessa parola. Inoltre questo stesso dizionario deve essere usato per implementare il grafo dei link per il calcolo del pagerank.

Dizionario delle parole

Una volta completato il dizionario delle pagine web, dovremo invertirlo per costruire un dizionario di tutte le differenti parole cha occorrono in tutte le pagine analizzate. Tale dizionario, implementato anch'esso con una tabella hash, deve contenere per ogni parola vista, una lista di riferimenti agli ipertesti in cui quella parola occorre almeno una volta.

Analisi di una query

Una volta completato anche il dizionario delle parole e calcolato il pagerank, saremo in grado di cercare una o più parole nelle pagine web delle directory di lavoro e restituirle ordinatamente secondo il rank calcolato.

Struttura del Progetto

Il programma verrà strutturato in differenti moduli descritti qui di seguito ad alto livello. Le caratteristiche specifiche di ogni modulo sarrano di volta in volta ampliate e dettagliate.

1. Nel primo modulo eseguiremo una scansione della directory di lavoro e creeremo una struttura dati che implementi il dizionario delle pagine e consenta, in un secondo momento di inserire ed accedere alle informazioni riguardanti le parole che occorrono ipertesto.

2. Un programma fornito dai docenti si occuperà di analizzare il file sorgente di un ipertesto e di restituire la prossima parola (si veda nella grammatica precedente la definizione di [parola]) riconosciuta nel testo. Tale programma dovrà quindi essere usato nel secondo modulo per analizzare tutti i file presenti nella directory, creare il grafo dei link e contemporaneamente per introdurre le informazioni per creare le *hitlists* di ogni differente parola incontrata nella pagina web.

3. Terminata la scansione di tutti i file presenti, si useranno le informazioni del grafo dei link per calcolare il pagerank associato ad ogni pagina. Questo sarà il compito del terzo modulo

4. Il quarto modulo deve costruire il dizionario delle parole. Usando il dizionario delle pagine si dovranno identificare tutte le differenti parole presenti nel testo e per ognuna mantenere una lista di riferimenti a tutti gli ipertesti dove occorrono.

5. Il quinto modulo usera il pagerank, il dizionario delle pagine e quello delle parole per rispondere ad una query formata da una o più parole restituendo la lista ordinata secondo il rank dei file dove tale parola occorre.