Componenti connesse

Dato un grafo orientato $G = (V, E)$, due nodi x e y sono in relazione $\leftrightarrow$ se esiste sia un cammino da x a y sia un cammino da y a x. La relazione $\leftrightarrow$ è una relazione di equivalenza, le cui classi sono dette *componenti (fortemente) connesse* del grafo.

Dato un grafo, è possibile costruire il grafo ridotto G^* i cui nodi rappresentano le componenti connesse del grafo e si ha un arco fra la componente C_i e la componente C_j se esiste un arco che va da un nodo di C_i a un nodo di C_j.

Struttura del Web

Il Web è una collezione di documenti. Questi documenti possono essere in vari formati, ma di preferenza sono ipertestuali e contengono collegamenti, detti link, verso altri documenti.

Si può pensare all'insieme dei documenti presenti sul Web come ad un grafo in cui:

- i nodi rappresentano gli URL
- un arco fra un nodo x e un nodo y rappresenta la presenza di un link all'interno della pagina che corrisponde all'URL x che punta verso l'URL y.

Le dimensioni del grafo del Web sono enormi, anche se difficili da valutare.

La struttura del Web viene detta "a cravattino" ed è costituita dalle seguenti componenti:

- componente gigante, comprende circa il 30% delle pagine;
- componenti sorgente, puntano direttamente o indirettamente verso la componente gigante ma non sono raggiungibili da essa, rappresentano circa il 24%;
- componenti pozzo, raggiungibili dalla componente gigante ma da cui non è possibile tornare indietro (es. pagine senza link), rappresentano circa il 24%;
- componenti isolate, non raggiungibili dalla componente gigante e da cui non si raggiunge la componente gigante;
- tentacoli, collegamenti tra componenti sorgente e componenti pozzo che non passano per la componente gigante.

Crawler

Un crawler o spider è uno dei componenti fondamentali di un motore di ricerca. Il suo compito è quello di raccogliere pagine, compito che svolge visitando i diversi nodi del grafo del web.

Copertura e scelta del seme

La copertura è sicuramente il problema più importante. Anche i migliori motori di ricerca coprono infatti solamente un parte ridotta del Web.

La raccolta inizia a partire da una o più pagine, dette seme della raccolta. La scelta del seme influisce in maniera forte sulla copertura ottenuta dal crawler, ossia sull'insieme delle pagine che verranno visitate da questo. Se ad esempio si parte da una pagina facente parte della componente gigante, la maggior parte del web verrà visitata, se viceversa si parte da una pagina facente parte della componente pozzo la copertura sarà molto più ridotta.

Scelta dei contenuti

La scelta dei contenuti da salvare è un altra scelta particolarmente problematica in questo ambito in quanto particolarmente influente sulla quantità di spazio su disco richiesto. Ad esempio, si potrebbe scegliere di salvare l'intero contenuto di ogni URL visitato, compresi gli header HTTP. Si potrebbe invece scegliere di salvare solamente il testo contenuto in una pagina oppure solamente la sequenza delle parole presenti all'interno di questa, escludendo tag e punteggiatura.

A volte non è necessario che il crawler visiti l'intero Web, ma solamente una parte di questo che un particolare criterio. Si potrebbe ad esempio volere che il crawler visiti solamente il Web italiano. Questo perà necessità la definizione di criteri che condizionino il processo di visita del crawler, criteri che ad esempio possono indicare quali URL seguire.

Politeness

Un altro problema è rappresentato dalla "politeness" richiesta al crawler. È necessario cercare di evitare il bombardamento degli host con troppe richieste consecutive, rispettando eventuali black listing e fornendo informazioni (user-agent) che consentano ai responsabili dei siti di sapere a chi rivolgere le proprie eventuali lamentele.

Strategia di visita

La strategia di visita stessa è un aspetto particolarmente critico, rispetto al quale è necessario prendere diverse decisioni, ad esempio: come selezionare la prossima pagina da visitare, si preferisce una visita in profondità o una visita in ampiezza?* Queste scelte hanno un forte impatto sulla rapidità con cui si raggiungono le pagine di alta qualità.

* per crawler di tipo generale (non focused) la visita in ampiezza risulta essere più efficiente.

Reazione alle anomalie

Un altro aspetto fondamentale è rappresentato dalla capacità del crawler di reagire alle anomalie. Un crawler deve essere robusto e tollerare le diverse anomali che possono proporsi durante la visita del Web (server che implementano HTTP in maniera scorretta, pagine HTML malformate, ...).

Crawling e tempo

Il crawler descritto finora è in grado di catturare un'istantanea del Web. Un ulteriore problema è però rappresentato dal fatto che il Web cambia nel tempo, con una frequenza significativa. Un vero crawler deve mantenere la sua rappresentazione del Web aggiornata nel tempo, andando a rivisitare le pagine già incontrate periodicamente. Questo ovviamente nei limiti definiti dalle norme di "politeness".

Modalità di crawling

Un crawler centralizzato può essere estremamente inefficiente nel caso in cui passi la maggior parte del tempo in attesa di I/O. Una soluzione è quella di fare crawling in modo multiprocesso o distribuito. Spesso un crawler è entrambe le cose:

- è distribuito, ossia costituito da tanti agenti, ognuno dei quali effettua una parte del crawl, visitando una porzione degli URL da visitare;
- è multiprocesso, ossia all'interno di ciascun agente ci sono molti processi che effettuano visite parallele, o che svolgono altre funzioni.

In un crawler distribuito si presentano tutti i problemi descritti in precedenza, e in più alcune problematiche nuove.

Crawling centralizzato

Come detto, un crawler può essere distribuito o centralizzato. Nel caso di crawler centralizzati, si definisce un coordinatore centrale che controlla il modo in cui procede il crawl. Questo coordinatore tiene tracci di quali URL sono già stati visitati e decide a chi assegnare ogni nuovo URL, eventualmente anche sulla base del carico di lavoro. Ogni agente, dopo aver scaricato la pagina a lui assegnata, dovrà comunicare i link al coordinatore centrale. I vantaggi di una soluzione di questo tipo sono rappresentati dal fatto che non si ha alcuna sovrapposizione tra le parti di web visitate dai diversi agenti che compongono il crawler, è possibile implementare delle politiche di ottimizzazione della ripartizione del carico e il numero degli agenti può cambiare nel corso del crawling. Gli svantaggi sono rappresentati dal grande scambio di informazioni richiesto e dal fatto che il coordinatore centrale rappresenta un collo di bottiglia e un single-point-of-failure per il sistema.

Crawling fully distribuited anarchico

In maniera del tutto opposta, gli agenti potrebbero essere del tutti liberi, ogni agente parte da un certo seme e procede indipendente, senza alcuna coordinazione con gli altri. In questo caso, si parla di fully distributed crawling di tipo anarchico. I vantaggi di una soluzione di questo tipo sono rappresentati dall'assenza di colli di bottiglia e single-point-of-failure oltre che dal non dover supportare una comunicazione fra gli agenti. Gli svantaggi di questo tipo di soluzione sono rappresentati dal fatto che tipicamente, molti agenti effettuaranno il crawling di porzioni di Web parzialmente sovrapposte e, inoltre, adottando una soluzione di questo tipo risulta particolarmente difficile realizzare una politica di politeness.

Crawling fully distribuited con assegnamento statico

In un crawler fully distributed con assegnamento statico non è presente un coordinatore centrale e l'insieme U degli URL da visitare viene partizionato a priori in n sottoinsiemi, uno per agente. Ogni volta che vinee etrovato un URL $u \in U_i$, questo viene comunicato all'agente i che ne è interamente responsabile. I vantaggi di una soluzione di questo tipo è che le sovrapposizioni citate in precedenza non si presenteranno più e che si semplifica in modo significativo l'implementazione di una politica di politeness. Viceversa, gli svantaggi di questa soluzione sono rappresentati dal fatto che, se un agente smette di funzionare, l'intera porzione di Web a lui assegnata viene persa. Inoltre risulta impossibile cambiare l'insieme di agenti funzionanti senza ricominciare da capo. Un ultimo problema di questa soluzione è che risulta impossibile controllare a priori il carico degli agenti, si può solo sperare che a cardinalità uguali corrispondano carichi uguali.