UNIVERSITÀ DEGLI STUDI DI TORINO

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
DIPARTIMENTO DI INFORMATICA

Progettazione e implementazione di un algoritmo PageRank per Arsmeteo

Relatore: Prof. Andras Horvath

Co-relatrice: Prof.sa Viviana Patti

Candidato

Fabio Bonsignore

a.a. 2007-2008

A Papà, Mamma, Titti e Filippo.E un ricordo affettuoso per i miei nonni.

Ringraziamenti

Desidero ringraziare per primi i miei genitori, che mi hanno sempre e a lungo sostenuto, anche quando questo ha significato per loro immensi sacrifici.

Uno speciale ringraziamento va ai miei relatori: il Prof. Horvath e la Prof.ssa Patti. Lavorare con loro si è dimostrato costantemente entusiasmante e stimolante. Sono stati disponibili e pazienti in ogni momento e mi hanno sempre messo a mio agio permettendomi di affrontare questo impegno con la maggior serenità possibile.

Ringrazio tutti i "ragazzi" di ArsMeteo ed Easybit: in particolare l'Architetto Giorgio Vaccarino, Flavio Portis ed Enrico Tenivella.

Ringrazio i tecnici, primo fra tutti Luca Boero per la sua amicizia e disponibilità.

Ringrazio il personale della portineria, soprattutto Giovanni, sempre paziente quando si avvicina l'ora di chiusura e qualche studente si attarda a lasciare i laboratori.

Abstract

In questi ultimi tempi nella rete è diventata sempre più evidente la coesistenza di una parte “vecchia” del web, statica e passiva, con una parte “nuova”, dinamica e attiva. Le due parti differiscono anche nel modo in cui gli utenti interagiscono con il web stesso e con gli altri utenti. Una caratteristica emergente è rappresentata dalle folksonomie, termine utilizzato per indicare un metodo di classificazione che a differenza delle tassonomie (dove il lavoro viene svolto da esperti) coinvolge persone non necessariamente preparate.

Alle informazioni presenti nella rete viene attribuito un valore numerico, detto rank, che ne indica l’importanza. Grazie a questo valore i motori di ricerca sono in grado di restituire in tempi più che accettabili risultati che sono rilevanti rispetto alle richieste degli utenti. In presenza di folksonomie il metodo di calcolo del rank deve essere modificato.

In questa tesi è stata investigata la possibilità di fornire una misura di rilevanza alle pagine descrittive degli elementi di un archivio sulla base della folksonomia associata, considerando come caso di studio il portale web ArsMeteo per la condivisione di materiale artistico multimediale. E’ stato progettato e implementato un algoritmo che valuta la rilevanza di ogni risorsa contenuta nel portale utilizzando l’informazione sui tag associati alle risorse. L’algoritmo si basa sui principi e sul funzionamento di PageRank, che stato adattato in modo da tenere conto della struttura di navigazione basata su tag del portale ArsMeteo.

Sommario

Indice delle tabelle

Indice delle figure

Indice delle appendici

Introduzione

Il contenuto della tesi potrebbe essere suddiviso in due parti. La prima comprende i capitoli dal primo al quinto e riguarda il PageRank: si inizia da alcune considerazioni su Internet e da una descrizione dei motori di ricerca per poi giungere alla matematica sulla quale si basa il calcolo, ad alcuni cenni sulla fattibilità, a degli esempi e ad alcune conseguenze importanti. La seconda comprende i capitoli dal sesto al nono e riguarda quello che si può chiamare FolkRank: si esaminano alcuni contesti in cui una nuova tendenza di contrassegnare le informazioni presenti nel web mediante etichette ha portato alle folksonomie, per poi descrivere il portale ArsMeteo e infine presentare le classi java dell'applicazione prodotta.

Il primo capitolo è una breve introduzione ai motivi per cui nacque la rete e al suo successivo sviluppo che giustificò la nascita di diversi tipi di motori di ricerca e di diversi modi di organizzare e reperire le informazioni; vengono esaminate le principali componenti dei motori di ricerca e si introduce Google e la sua caratteristica principale.

Il secondo capitolo tratta degli aspetti matematici del calcolo del PageRank secondo Google (diversi motori di ricerca possono adottare diversi criteri di valutazione) e mostra come si sia trovata soluzione ad alcuni ostacoli che impedivano il calcolo.

Il terzo capitolo descrive la difficoltà principale che può essere riscontrata quando si voglia eseguire il calcolo nella maniera in cui è stato presentato nel secondo capitolo; in base a queste difficoltà e con altra poca matematica la formula generale viene trasformata in maniera che possa essere eseguita con una minore richiesta di risorse.

Il quarto capitolo comprende alcuni esempi pratici in cui viene mostrato effettivamente cosa succede quando le parti coinvolte nel calcolo non hanno tutte le caratteristiche richieste.

Nel quinto capitolo viene affiancato il concetto di PageRank alla progettazione di siti web, facendo vedere come con una buona progettazione possa ottenersi una valutazione migliore della propria homepage: vengono anche accennati i problemi e le conseguenze a cui si va incontro se per lo stesso scopo si utilizzano tecniche moralmente e giuridicamente perseguibili.

Il sesto capitolo è un'analisi di una nuova "moda" presente da pochi anni nel web: le risorse incontrate durante la navigazione (testi, grafici, audio, video,…) possono essere descritte mediante delle metainformazioni: questo fenomeno ha portato a diverse ramificazioni del web.

Il settimo capitolo si occupa di una nuova attività di catalogazione e classificazione di un insieme di elementi originatasi dall'abitudine di etichettare le risorse presenti nel web: le folksonomie (contrapposte alle tassonomie).

L'ottavo capitolo riguarda la struttura del portale ArsMeteo e ne descrive le caratteristiche principali.

Il nono capitolo descrive l'applicazione sviluppata per attribuire i valori di FolkRank alle opere presenti in ArsMeteo: sono presentate le classi java così come documentate in stile javadoc a cui sono aggiunte ulteriori spiegazioni per quelle parti che sono state più impegnative.

Infine l'appendice comprende qualche richiamo riguardante alcune notazioni e definizioni che si incontrano nei calcoli che coinvolgono matrici.

Capitolo 1
Reperire informazioni dal web: i motori di ricerca

Uno dei vantaggi dell'utilizzo di Internet è stato un nuovo modo di concepire la comunicazione, l'organizzazione e la presentazione di informazioni: la definizione del protocollo **HTTP** e del linguaggio di annotazione **HTML** insieme alla produzione di *browser*, hanno reso possibile la fruizione di una grande quantità di contenuti ad un gran numero di persone.

La quantità di documenti presenti nel *web* è enorme ed è necessario un modo di concepire la ricerca di informazioni che sia adatto e scalabile: i motori di ricerca, via via sviluppandosi e introducendo nuove caratteristiche, soddisfano più che sufficientemente le aspettative degli utenti. A qualche anno dalla nascita del primo motore di ricerca, tra quelli presenti si distingue *Google* che in poco tempo ha guadagnato il monopolio nell'ambito della ricerca nel *web* (sembra di riconoscere il modello della selezione naturale nella teoria dell'evoluzionismo di Darwin: in questo contesto il motore più funzionale ha avuto un vantaggio competitivo rispetto agli altri).

1.1 Introduzione

All'inizio degli anni sessanta Internet non era che una piccola idea appena abbozzata, non ben definita e senza dubbio molto diversa dalla realtà odierna. Da un progetto della DARPA (*Defense Advanced Research Project Agency* – Dipartimento della difesa statunitense)[1] l'intenzione era quella di sviluppare una rete telematica decentrata che potesse garantire le comunicazioni in ogni caso (era il triste periodo della guerra fredda e in particolare si voleva che anche nel caso di attacchi con conseguente distruzione di obiettivi come i centri di trasmissioni dell'esercito, fosse possibile

[1] In una nota del 25 aprile 1963, Joseph C. R. Licklider, informatico statunitense prese in considerazione la possibilità di collegare tutti i computer in una rete continentale.

indirizzare i comunicati attraverso percorsi alternativi, per aggirare i problemi causati dai reparti guastatori nemici). Una successiva espansione si ottenne collegando i principali centri universitari e di ricerca e poi garantendo usi civili, fino a raggiungere l'utenza aziendale e quella domestica.

Nel periodo 1989-1991 presso il CERN di Ginevra il ricercatore Tim Berners-Lee, l'ingegnere informatico francese Robert Caillau e altri scienziati europei e americani, definiscono: il protocollo **HTTP** *(HyperText Transfer Protocol)* che permette una lettura non sequenziale di documenti tramite rimandi (*hyperlink*, detti semplicemente *link*), i *browser* (programmi specializzati che consentono agli utenti di visualizzare e interagire con testi, immagini e altre informazioni contenute nelle pagine o nei siti) e il linguaggio **HTML** (per la composizione di documenti ipertestuali); nasce il *World Wide Web*[2] (il primo *browser*, Mosaic, con caratteristiche simili a quelle degli attuali, venne realizzato nel 1993). **HTTP** e *browser* rendono possibile l'uso di Internet a decine di milioni di persone (soprattutto a quelle al di fuori dell'ambito informatico): ogni utente (sia esso persona fisica o no) ha la possibilità di pubblicare centinaia di pagine.

Allo stato attuale il web è costituito da centinaia di milioni di documenti (8.1 miliardi, secondo stime del 2006) ed avere così tanta informazione equivale paradossalmente a non averne. Si paragona sempre più il *web* alla biblioteca di Babele[3]: un luogo di fantasia in cui sono raccolti tutti i possibili libri (dove *"perché un libro esista, basta che sia possibile"*): tutti quelli scritti e anche tutti quelli che sarebbe possibile scrivere, in qualunque lingua in qualunque periodo; non solo: in questa biblioteca chiunque può entrare e depositare un libro, un suo manoscritto, una poesia e quant'altro, il tutto semplicemente passeggiando e collocandolo nel primo scaffale che trova, senza un codice che possa identificare il materiale depositato. Non essendoci

[2] Tim Berners-Lee, L'architettura del nuovo Web - Dall'inventore della rete il progetto di una comunicazione democratica, Feltrinelli, Milano, 2001

[3] In un racconto di Luis Borges, scrittore, narratore, saggista e poeta argentino. Famoso per i suoi racconti fantastici, in cui ha collegato filosofia e metafisica con i temi del fantastico (quali: il doppio, le realtà parallele del sogno, i libri misteriosi e magici, gli slittamenti temporali). Oggi l'aggettivo «borgesiano» è sinonimo di menzogna (famose le sue recensioni di libri immaginari).

nessun catalogo [4], nessun ordinamento, trovare qualcosa diventa un'impresa enorme. Inoltre in Internet le risorse sono dinamiche (vengono inserite, possono essere aggiornate nel tempo o venire cancellate) oppure ancora l'oggetto del cambiamento nel tempo sono le connessioni che permettono di raggiungere le risorse: si intravede la necessità di un cambiamento di metodo nel modo di gestire (e ricercare) le conoscenze disponibili.

L'ideale è un servizio in grado di consentire tempi di risposta rapidi, non soltanto dipendentemente dal titolo ma anche dal contenuto. Questi servizi sono quelli offerti dai motori di ricerca che per quanto efficienti non possono però impostare i criteri di ricerca per l'utente: è quest'ultimo a doverlo fare (in modo corretto).

1.2 Motori di ricerca

Per cercare qualcosa, non serve soltanto lo strumento adatto: bisogna anche saperlo utilizzare. E' chi cerca che deve ottimizzare, altrimenti anche il miglior motore può dare risultati insoddisfacenti.

Vi sono due principali tipi di ricerca: quella per categorie e quella per parole chiave.

Nella ricerca per categorie (o canali o soggetti) ci si basa su un progressivo raffinamento di ciò che si cerca, partendo da categorie molto ampie e generali per giungere ad argomenti più specifici.

Nella ricerca per parola chiave si fornisce un testo che si pensa possa essere contenuto nel documento che si sta cercando: il caso più semplice è quello di una sola parola, ma questo funziona con parole particolari ben definite e senza ambiguità o sinonimi. In questo tipo di ricerca si procedere in diversi modi:

in sequenza: ma solo se il documento è piccolo (in un elenco telefonico, per esempio, non avrebbe tanto senso);

[4] Se anche ci fosse, un catalogo del genere sarebbe limitato dalle sue stesse dimensioni: la ricerca in un indice di un miliardo di titoli è un'attività onerosa per un essere umano (senza contare il fatto che il catalogo non si saprebbe dove cercarlo nella suddetta biblioteca).

indicizzazione: si ispira al concetto di indice analitico di un libro (un lettore cerca la parola nell'indice e si sposta nella pagina indicata dall'indice); i motori di ricerca visitano periodicamente i siti costruendosi degli indici: quando viene loro indirizzata una richiesta di ricerca consultano i file di indice che si sono costruiti e restituiscono i riferimenti trovati;

ranking: si costruisce una gerarchia di documenti cercando di scoprire quali possano essere più interessanti tramite una assegnazione di punteggio (diversi motori di ricerca possono avere diversi criteri di ranking);

variazioni della ricerca per parola chiave: per ottimizzare, al posto della parola si mettono delle frasi (tra doppi apici), oppure si fa uso di operatori booleani (and, not, or), di operatori di prossimità (near) e di ricerche cosiddette elastiche (la radice di una parola seguita da un carattere jolly); gli operatori booleani raramente vengono usati dall'utente non esperto;

ricerche avanzate: una combinazione delle precedenti; anch'esse raramente utilizzate dall'utente non esperto.

Vengono offerti spesso servizi gratuiti aggiuntivi come e-mail, meteo, pagine gialle e bianche, people search, shopping e altri.

I motori di ricerca possono essere suddivisi in alcuni gruppi:

generalisti (**altavista, yahoo, hotbot, excite, lycos, google**);

dedicati (**arianna, digiland, aziende.it, il trovatore, Virgilio**)

finalizzati (**tucows, FTPSearch, drivershq.com, Astalavista, Pronto.it**)

metamotori: i motori esistenti coprono solo una parte del web; i metamotori si occupano di interrogare più motori per ottenere risposte più precise e sono consigliati per ricerche esaustive (**metacrawler**);

e altri ancora, utili per la ricerca di parole, dizionari, spettacoli, viaggi, musei virtuali, astronomia, medicina e cultura in genere.

La lista non si esaurisce certo qui.

I motori che permettono di ricercare le informazioni nella rete sono in realtà un po' più vecchi del *web*, perché sin dall'inizio si sentì la necessità del reperimento di informazioni: già da anni i servizi di ricerca erano operativi, ma non erano consultabili via *web* ed erano programmi da installare e configurare nella propria macchina.

E' del 1994 il primo motore di ricerca basato su web: **WebCrawler**, nato da un esperimento finanziato dall'università di Washington (esiste ancora, ed è un ottimo "metamotore" di ricerca, in quanto combina i risultati dei migliori motori). Funzionava come un sito Internet e offriva la possibilità di effettuare ricerche *fulltext*, (parole singole, combinazioni di parole, frasi complete) scorrendo i documenti nella loro completezza; era dotato di uno *spider*[5]. Nonostante questo, tuttavia, era solo in grado di fornire risultati costituiti da semplici elenchi di indirizzi *web* (e titoli di pagine).

Intorno alla fine dello stesso anno seguì **Lycos**, la cui innovazione era quella di analizzare soltanto le prime 20 righe di ogni pagina, invece che l'intero contenuto.

L'anno successivo fu la volta di **Excite**, i cui risultati delle ricerche venivano ordinati in base a un criterio di importanza.

Nel 1997 il posto di miglior motore di ricerca spettava ad Altavista: utilizzava un database per la consultazione, mediante *cluster* di computer[6] collegati fra loro, hardware all'avanguardia e il meglio delle tecnologie allora disponibili. E' stato anche il primo motore di ricerca multi-lingua di Internet.

1.3 Elementi dei motori di ricerca

Diverse parti contribuiscono a far funzionare un motore di ricerca.

---

[5] Strumento per catalogare le pagine automaticamente: è un software che mentre naviga in Internet memorizza le informazioni contenute sulle pagine e le rende accessibili agli utenti del servizio di ricerca.

[6] Un **cluster** (dall'inglese *grappolo*) di computer è un insieme di computer connessi tramite una rete con lo scopo di distribuire una elaborazione molto complessa tra i vari computer componenti il cluster per aumentare la potenza di calcolo del sistema.

modulo di scansione[7]: è la parte di *software* incaricata della creazione di *spybot* (altro modo di chiamare gli *spider*) che si occupano di scandagliare il *web* con lo scopo di raccogliere e classificare i documenti trovati e conservarli in un "deposito"; è utile perché nel *web* non esiste nessuna struttura fisica centralizzata che raccolga e limiti la grandezza delle pagine;

deposito di pagine: conserva le pagine provenienti dagli spider fino al momento in cui esse vengono trasferite per l'operazione di indicizzazione;

modulo di indicizzazione: le pagine provenienti dal deposito di pagine vengono compresse e memorizzate in vari indici;

indici: conservano le descrizioni date ad ogni pagina dal modulo di indicizzazione e hanno scopo simile a quello degli indici che si trovano in fondo ai libri; possono essere di tre tipi (di contenuto: memorizzano informazioni come parole chiavi, titolo e testo; di struttura: contengono informazioni riguardanti i collegamenti delle pagine; di specializzazione: mantengono informazioni per scopi speciali e riguardano file particolari, come immagini, pdf, doc,…);

query: la richiesta fatta dall'utente viene convertita in una forma comprensibile al motore di ricerca (nei metamotori questa trasformazione è più curata in quanto il metamotore dialoga con più motori e deve fare in modo che tutti possano capire la *query*); il modulo di *query* consulta gli indici di contenuto per trovare quali pagine sono interessanti per l'utente (l'insieme delle pagine rilevanti è poi passato al modulo di *ranking*);

modulo di *ranking*: classifica le pagine rilevanti (ricevute dal modulo precedente) in base ad alcuni criteri; il risultato è una lista contenente pagine *web*, in cui quelle "in cima" sono le più utili per l'utente.

1.4 Google e PageRank

In luoghi separati il fenomeno della crescita esplosiva di Internet attrae e affascina due studenti: destino vuole che essi si incontrino nel 1995 all'università di

[7] Uno scansionatore può produrre più spider

Stanford. Il primo, Larry Page, in visita con l'intenzione di iscriversi al dottorato di informatica e il secondo, Sergey Brin, appartenente al gruppo di studenti assegnati all'accompagnamento dei visitatori del campus.

Quello che Larry Page ha in mente, è di trovare un qualche modo per calcolare l'importanza di una data pagina del web. Un aiuto alle sue idee viene dato dalle ricerche del matematico e statistico russo Andrej Andreevic Markov nei primi del Novecento, riguardante l'importanza di un nodo di una rete rispetto ad altri della stessa.

Secondo Page, una valutazione di una data pagina può essere ricavata considerando il numero di link che portano alla pagina stessa (un tale valore può anche indicare la bontà, o qualità, del documento). Ma considerare solo il numero di link non è sufficiente: i link stessi non sono uguali e non hanno la stessa importanza. Infatti, un sito molto popolare che "indica" una pagina, dà a quest'ultima un contributo di importanza maggiore rispetto a quello di un altro sito meno importante (in altri termini, il "giudizio" di una pagina molto popolare conta più del "giudizio" di una pagina poco nota). Questo metodo di valutazione delle pagine viene detto *PageRanking*.

Un anno dopo il suo incontro con Brin, Page incorpora l'algoritmo nel suo spider e lo esegue: di lì a poco i primi risultati. Si notò però un eccessivo consumo di banda, problema significativo per gli amministratori di rete alla Stanford[8]. Oltre a questo, lo spider dovette cominciare a fare i conti con il fatto che si presentarono due ostacoli: da una parte i detentori dei siti si preoccupavano dei diritti di copyright dei propri materiali e dall'altra si era preoccupati dal fatto che questo metodo di ranking non tenesse affatto conto dei premi ufficiali o di altri sistemi di valutazione consolidati: lo spider era totalmente estraneo ai contenuti delle pagine. Questa caratteristica è anche un punto di forza su cui si baserà il successo di Google, in quanto i risultati non sono più ottenuti in maniera definitiva una volta per tutte, ma assumono valori diversi in diversi intervalli temporali dipendentemente dalle connessioni delle pagine.

[8] Gli spider devono minimizzare questo problema: il "protocollo di esclusione dei robot" è stato definito appunto per limitare le attività degli spider e per punirne attività non corrette (gli amministratori dei siti possono utilizzare un file denominato "robot.txt" per bloccare l'accesso a parti dei loro siti).

Il metodo viene testato per un anno e mezzo, facendo uso solo di strumenti liberi la cui disponibilità è data dalla comunità Free Software e Open Source. Questa è un'altra delle ragioni del successo di Google: gli strumenti liberi permisero la costruzione di uno strumento modificabile (modulare) ed espandibile (scalare) anche a pieno regime di utilizzo. Questi pochi e relativamente semplici strumenti sono alla base del Googleplex, il *datacenter* di Google (composto da vari *datacenter* minori) che si occupa di elaborare le richieste ricevute, di propagare eventuali modifiche, di sostituire, aggiungere e manutenere software ed hardware senza mai dover bloccare i servizi. La decentralizzazione causa però disallineamenti che vengono risolti entro pochi giorni (si parla di *Google Dance* intendendo la variazione dei valori calcolati dal *pageranking*).[9] I *datacenter* hanno forti caratteristiche di ridondanza per evitare (o limitare) la possibilità di perdite e contemporaneamente ottenere maggiore efficienza.

In queste condizioni (e grazie ai fallimenti seguiti al periodo del dot commerce) agli inizi del 2000 Google trova una strada sgombra o con pochissimi concorrenti e può cominciare indisturbato con la cura dei propri utenti tramite l'offerta di molti nuovi servizi: ricerca di immagini, blog, notizie, mappe, video, finanze. Già nel 2002 Google conta oltre 1000 dipendenti, 10000 computer e 4 miliardi di pagine esaminate e vengono proposte altre offerte: servizi comunicazione (email, voip, messaggistica istantanea, gruppi di discussione, condivisioni di fotografie, servizi di traduzione), servizi riguardanti dispositivi portatili (cellulari, PDA), servizi riguardanti software e altri ancora, insieme agli interessanti Google Maps per la Terra, Marte o per la Luna (2005 e 2006), Google News, Google Scholar e Google Library.[10]

[9] http://www.motoricerca.info/articoli/google-dance.phtml
[10] http://www.google.com/intl/en/options/ per una lista di prodotti

Figura 1.1 - La pagina di ricerca di Google così come viene mostrata dal browser mozilla versione 2 e, sotto, Larry Page (a sinistra) e Sergey Brin.

1.5 Il modello di Google

Google ha dunque raggiunto un grande successo affermandosi come il motore di ricerca più utilizzato in Internet: qualche volta si sentiva dire: “Se non lo sai, chiedilo a Google” (qualcuno si è anche riferito a Google con l’appellativo di “verbo” per la ricerca sul web).

A volte si dice che il modello di Google è un modello, “democratico” perché l’algoritmo utilizzato interpreta un collegamento da una pagina ad un’altra come una sorta di voto espresso dalla prima a favore della seconda. In generale si tratta di considerare voti provenienti da più pagine.

In realtà, nel far questo, il procedimento tiene conto anche di un altro fattore e quindi non è una vera e propria democrazia: come già accennato, i voti assegnati dalle pagine che hanno una certa importanza, apportano un valore maggiore rispetto ai voti assegnati da pagine che hanno minore importanza (un’altra cosa da dire è che il calcolo

non tiene conto del contenuto o della grandezza delle pagine: non dice nulla ad esempio sulla lingua in cui sono scritte o se contengono immagini, audio o video).

C'è da notare infine una nuova tendenza, nata recentemente e di cui si parlerà nel seguito, per cui alcuni siti mettono a disposizione degli utenti un modo per etichettare le risorse presenti nel sito stesso: questo da' vita a un nuovo modo di ricercare che è più orientato all'utente non esperto e promettente per quanto riguarda contenuti multimediali.

Vedremo in dettaglio sia i meccanismi che permettono l'attribuzione del pagerank alle pagine, che i meccanismi che agiscono nel caso di "tag".

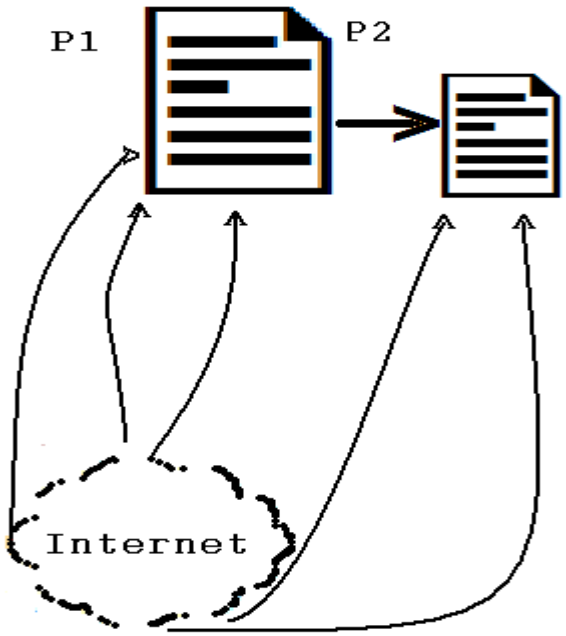

Figura 1.2 - Il modello di attribuzione del voto: le pagine più importanti forniscono un contributo maggiore rispetto a quelle meno importanti.

Capitolo 2
Aspetti matematici del PageRank

Il meccanismo che consente l'attribuzione del punteggio alle pagine web si basa su poche assunzioni e poca semplice matematica, sebbene le formule possano sembrare complicate in un primo momento.

Dalle considerazioni matematiche di base si giunge al metodo che permette il calcolo dei valori delle pagine; tale metodo richiede però delle caratteristiche che non si riscontrano in alcuni elementi presenti nel calcolo stesso: il modello del navigatore immaginario permette di superare questi ostacoli.

2.1 Premessa: cenno al modello del navigatore immaginario

Questo modello approssima il comportamento di un utente reale che naviga nel web (da cui l'appellativo "immaginario"): il navigatore esegue il browser e inizia la navigazione, seguendo i collegamenti presenti, passando di pagina in pagina. E' questo modello che ha aiutato gli ideatori dell'algoritmo nella realizzazione del loro progetto e che permetterà di esporre più facilmente alcuni dettagli.

In un primo momento, il modello rappresenta una certa probabilità che il navigatore stia su ogni pagina che visita per la stessa quantità di tempo, continuando a navigare senza fermarsi, passando da link a link e impiegando lo stesso intervallo di tempo per ogni visita. In questa sua prima forma, il modello non descrive alcune situazioni che l'utente può incontrare durante il suo viaggio.

La descrizione di questo modello sarà quindi completata e rafforzata nel seguito, quando sarà opportuno fare alcune precisazioni: in particolare, non si era tenuto conto del fatto che un navigatore reale può comportarsi in maniera leggermente diversa dal navigatore immaginario; quando si incontrarono gli ostacoli che vedremo, si cominciò anche ad osservare il comportamento dei navigatori veri e propri.

Ecco i concetti matematici che permettono il calcolo del PageRank.

2.2 Matematica del PageRank

Protagonisti del calcolo sono alcune somme e poche operazioni e proprietà vettoriali e matriciali (riportate in appendice).

Notazioni utilizzate:

PR abbreviazione di PageRank

$P_j \longrightarrow P_i$ la pagina P_j contiene un collegamento alla pagina P_i

$B_{P_i} = \{P_j / P_j \longrightarrow P_i\}$ insieme delle pagine P_j che puntano a P_i

$\left|P_j\right| = n \in \mathrm{N}$ numero di collegamenti uscenti dalla pagina P_j

$$r(P_i) = \sum_{P_j \in B_{P_j}} \frac{r(P_j)}{\left|P_j\right|}$$ PR della pagina P_i

Come si vede dall'ultima relazione, il PR di una pagina P_i dipende dal PR delle pagine P_j che puntano a P_i; ciò significa che prima si dovrebbe calcolare il PR delle pagine P_j e prima ancora il PR di quelle che puntano a P_j (e così via) perché anche gli $r(P_j)$ sono sconosciuti.

TUTTAVIA è possibile calcolare il PR di una pagina SENZA conoscere il valore FINALE del PR delle altre, per "approssimazioni[1] successive", semplicemente ricordando i valori ottenuti a partire da una approssimazione iniziale e ripetendo i calcoli con i nuovi valori, fino a quando la differenza fra il risultato di una iterazione

[1] I termini: approssimazioni, raffinamenti, iterazioni, passi sono equivalenti

precedente e il risultato della iterazione attuale non è minore di una certa quantità (ossia finché non si raggiunge la precisione voluta).

Il PR della pagina P_i al passo (k+1) si esprime quindi con:

$$r_{k+1}(P_i)$$

La formula del calcolo del PR diventa[2]:

$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_j}} \frac{r_k(P_j)}{|P_j|}$$

L'approssimazione iniziale viene stabilita impostando:

$$r_0(P_i) = \frac{1}{n}$$

per tutte le pagine, dove n è il numero delle pagine considerate e la frazione (quella in cui compare il denominatore $|P_j|$) è il contributo di ogni pagina; tale contributo non è il contributo "totale", perché se vi sono più collegamenti uscenti da una pagina, il contributo deve essere diviso per il numero di collegamenti uscenti. La parte centrale della seguente espressione dice proprio questo (e VALE SOLO INIZIALMENTE):

$$r_1(P_i) = \sum_{P_j \in B_{P_j}} \frac{r_0(P_j)}{|P_j|} = \sum_{P_j \in B_{P_j}} \frac{\frac{1}{n}}{|P_j|}$$

Facendo così, però, il PR viene calcolato una pagina alla volta, mentre si vorrebbe poterlo valutare per tutte le pagine ad ogni passo. E' a questo punto che si cominciano ad utilizzare matrici e vettori: si può infatti sostituire la sommatoria con una

[2] Per la convergenza si veda nel seguito

matrice e, per ogni iterazione, calcolare il PR di tutte le pagine; a questo scopo si denotano con:

p^k matrice di dimensioni (1,n) dei valori $r(P_i)$ all'iterazione k-esima

H matrice di dimensione (n,n) dei collegamenti fra le pagine, definita come segue:

$$h_{ij} = \begin{cases} \frac{1}{|P_j|} & se \quad P_i \longrightarrow P_j \\ 0 & altrimenti \end{cases}$$

Questa matrice è la matrice di adiacenza delle pagine, dove sono rappresentati tutti i collegamenti che dalla pagina i giungono alla pagina j, per tutte le pagine considerate; in essa gli elementi diversi da zero sono delle probabilità[3].

Nella matrice di adiacenza, gli elementi della i-esima riga, se diversi da zero, rappresentano gli archi USCENTI dalla pagina i; gli elementi della j-esima colonna, se diversi da zero, rappresentano gli archi ENTRANTI nella pagina j.

Eventuali righe costituite da tutti zero, rappresentano invece il fatto che da una data pagina P_i non è possibile raggiungere nessuna delle altre pagine considerate (questo è un limite del modello della navigazione del quale non si era tenuto conto inizialmente, perché è come se ad un certo punto si rimanesse imprigionati senza avere più nessuna possibilità di ritornare indietro o di continuare la visita delle pagine: da un

[3] La somma degli elementi di OGNI riga è uguale all'unità e gli elementi della riga rappresentano la probabilità che una pagina sia visitata.

punto di vista matematico restare intrappolati nella matrice equivale a non raggiungere nessun risultato).

Dunque, con la definizione di p e H la:

$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_j}} \frac{r_k(P_j)}{\left|P_j\right|}$$

assume la forma più leggibile e concisa:

$$p^{k+1} = p^{k} H$$

dove adesso p^{k+1} contiene TUTTE le r al passo k+1: ecco che più PR vengono calcolati in un "unico" passaggio; questa formula soffre però di alcuni punti deboli, perché si possono presentare[4]:

AFFOSSAMENTI DEL POSIZIONAMENTO (RANK SINKS): può succedere che alcune pagine accumulino sempre più PR ad ogni iterazione, monopolizzando la valutazione, rifiutando la condivisione e rendendo "non equo" il risultato; cioè se $p = (p_1 \quad p_2 \quad \dots \quad p_n)$, $n \geq 2$ può succedere ad esempio che: $p_i = 0 \quad per \quad i \leq \frac{n}{2} \quad e \quad p_i > 0 \quad per \quad i > \frac{n}{2}$;

OSCILLAZIONI: in presenza di cicli, non importa quanto grande diventi k, il risultato non converge; cioè i valori oscillano indefinitamente; ad esempio: $p^{2k} = (0 \quad 0 \quad \dots \quad p_n)$ e $p^{2k+1} = (p_1 \quad p_2 \quad \dots \quad 0)$, dove 2k e 2k+1 indicano rispettivamente un numero pari ed un numero dispari e dove gli elementi denotati da p_i sono diversi da zero

[4] Esempi numerici per ogni caso saranno trattati nel capitolo 4.

Quindi fin qui si riescono a calcolare più PR facendo uso delle matrici, ma possono presentarsi dei casi in cui o non si giunge a un risultato (cicli) o il risultato non è quello voluto (affossamenti).

In particolare si vuole che il vettore risultato sia un vettore positivo, ossia un vettore tale che ogni suo elemento sia strettamente maggiore di zero.

Osservando l'equazione, si nota che assomiglia ad un metodo di calcolo detto metodo delle potenze che viene applicato su una catena di Markov con matrice di probabilità di transizione P[5]; in tale calcolo, per qualsiasi vettore (positivo) di partenza, il metodo delle potenze applicato a P CONVERGE ad un UNICO vettore POSITIVO normalizzato a condizione che P sia STOCASTICA, IRRIDUCIBILE e APERIODICA (in questo caso si è sicuri che il risultato è quello voluto: nessuna oscillazione, nessun affossamento e vettore risultato positivo).

H possiede le caratteristiche richieste (o si può modificare o sostituire in modo che le abbia)?

E' a questo punto che si sfrutta nuovamente il concetto del navigatore e lo si rafforza (tenendo conto del fatto che le quantità in H rappresentano la presenza, o l'assenza, di link):

da una pagina con più link uscenti se ne sceglie uno a caso e si continua a navigare;

lasciando trascorrere lo STESSO intervallo di tempo, quindi periodicamente[6], regolarmente, tra un click e quello successivo, alcune pagine sono rivisitate: sono le più importanti perché maggiormente collegate alle altre;

le pagine senza link, dette pendenti, non vietano al surfer di scegliere un link a caso e continuare la navigazione (infatti, osservando un navigatore reale approdare ad una pagina pendente, questi, se decide di continuare la navigazione, "salta" a caso ad un'altra pagina); proprio questa osservazione suggerisce di adottare un accorgimento:

[5] Per il metodo delle potenze e per le catene di Markov vedere l'appendice.
[6] Come era il modello iniziale (ricordiamo che serve però NON la periodicità MA l'aperiodicità)

quelle righe di H che sono costituite esclusivamente da zeri (in questo caso H è detta "sottostocastica") NON permettono al navigatore di sfuggire dai nodi pendenti; per risolvere questo inconveniente, si conviene che una pagina pendente sia in realtà collegata a tutte le altre (conseguenza dell'osservazione del comportamento di un navigatore reale) e gli elementi delle righe zero di H diventano tutti elementi frazionari, i cui numeratori sono l'unità e i cui denominatori sono il numero di pagine presenti nell'insieme B_{P_i} (in questo modo la somma degli elementi della riga è ancora l'unità e viene rappresentata sempre una probabilità).

Si introduce così la matrice A, definita da:

$$a_{ij} = \begin{cases} \frac{1}{n} & se \quad R_i = 0 \\ 0 & altrimenti \end{cases}$$

dove la notazione $R_i = 0$ indica che la i-esima riga della matrice è composta da tutti zero.

La matrice $S = H + A$ costruita in questo modo è STOCASTICA (e quindi matrice di probabilità di transizione per una catena di Markov).

Bisogna ora completare il modello del navigatore, perché quello fin qui utilizzato non è ancora adeguato non tenendo conto di alcuni fatti:

- non è detto che il navigatore impieghi sempre lo stesso tempo tra un click e il successivo (aperiodicità)

- sempre osservando il comportamento di un navigatore reale, si è notato come a volte, per noia diciamo, anche se ci sono link uscenti dalla pagine in cui si trova, abbandona il metodo di navigazione a collegamento, per inserire una nuova destinazione (a caso) nella URL del browser (allora si può pensare che ogni pagina è direttamente collegata alle altre: questo accorgimento suggerisce l'irriducibilità).

La matrice finale a cui si arriva è detta matrice di Google di dimensioni (n,n) e denotata da

$$G = aS + (1-a)1_N$$

dove:

a è la percentuale di tempo in cui il surfer segue i link o salta dalle pendenti

$(1-a)$ è la percentuale di tempo in cui il surfer salta per noia[7]

1_N è la matrice di dimensioni (n,n) con elementi tutti uguali a $\frac{1}{n}$ (stocastica)

In questo modo si è ottenuta una matrice G stocastica e primitiva (aperiodica e irriducibile), adatta ad essere utilizzata nel metodo delle potenze. Proprio le caratteristiche di G assicurano la convergenza (e in particolare che i risultati saranno quelli voluti):

$$p^{k+1} = p^k G$$

dove le dimensioni di p e G sono del tipo richiesto per poter effettuare il prodotto.

[7] Questo addendo ci assicura anche che tutte le pagine avranno un voto minimo, comprese quelle che non hanno collegamenti entranti (cioè quelle per cui nessun'altra pagina esprime un voto).

Capitolo 3
Cenni ad algoritmo e complessità

Gli aspetti matematici visti nel capitolo precedente sono le fondamenta su cui si basa l'edificio per il calcolo del PageRank. Tuttavia non sono solo le fondamenta che fanno un immobile.

Chiariti i passi da fare per il calcolo, si possono realizzare programmi che lo effettuano, ma prima di progettarli e scriverli bisogna condurre un'analisi in modo da poter determinare a priori il quantitativo di risorse che saranno richieste dal calcolo.

Data la quantità di pagine presenti nel web, la risorsa che più preoccupa in questo contesto è la memoria.

3.1 – Questione di spazio

Con il termine "complessità in spazio" si indica la quantità massima di memoria richiesta durante l'esecuzione dell'algoritmo, il quale può costruire insiemi di dati intermedi o di servizio, oltre ad operare sui dati iniziali e finali.

E se la quantità di memoria a disposizione non è sufficiente l'esecuzione dell'algoritmo non è alla nostra portata.

Rappresentare tutte le pagine del web significa costruire una matrice di dimensioni (8.1E9, 8.1E9); le richieste di memoria sono assurde: se ad esempio si pensa di rappresentare gli elementi della matrice con il tipo di dato double java (che richiede 64 bit) sono necessari 64 x 65,61E18 bit, cioè 524,88E18 byte, cioè ancora, arrotondando per troncamento, 477.375.579 TB: è una quantità spaventosa!

E' chiaro adesso il perché del gran numero di computer necessari a Google e di cui si è accennato nel primo capitolo; tuttavia pur facendo in questa maniera il compito sarebbe arduo se non si sfruttassero dei piccoli accorgimenti.

3.2 – Un passo indietro

Nella formula

$$p^{k+1} = p^{k} G$$

ottenuta nel capitolo precedente la matrice G ha certe caratteristiche che non la rendono adatta (in termini di spazio di calcolo richiesto). Infatti G è completamente densa e questo è deleterio, computazionalmente parlando.

E' stata fatta tanta strada per ottenere le auspicabili proprietà di convergenza, ma ci si ritrova con una grande complessità computazionale. La domanda è se è possibile fare qualche passo indietro e riscrivere la formula in una maniera tale che si possano conservare le proprietà richieste e al contempo ottenere una forma che sia meno esigente in fatto di risorse.

La risposta è affermativa. A questo scopo riscriviamo alcune eguaglianze:

$A = a\frac{1}{n}e_R$ dove a è il vettore colonna in cui compaiono degli 1 in corrispondenza di quelle righe che in A sono pendenti ed e(R) è il vettore riga composta da tutti 1.

$1_N = \frac{1}{n}e_C e_R$ dove e(C) è il vettore colonna composto da tutti 1 ed e(R) è il vettore riga composto da tutti 1.

In questa maniera:

$$p^{k+1} = p^{k} G = p^{k}\left[aS + (1-a)1_N\right] = p^{k}\left[a(H+A) + (1-a)1_N\right] =$$

$$p^{k}\left[a(H + \frac{1}{n}ae_R) + (1-a)\frac{1}{n}e_C e_R\right] = p^{k} aH + p^{k}\left[aa + (1-a)e_C\right]\frac{1}{n}e_R$$

Così facendo G NON viene mai costruita (pur rimanendo le proprietà volute) e si memorizzano solo alcuni valori di H ed a (oltre al vettore dei risultati).

Ora, H è una matrice molto sparsa, perché la maggior parte dei suoi elementi è zero (nella realtà questo significa che la maggior parte delle pagine web contiene soltanto pochi link a qualche altra pagina). L'elemento che occorre il maggior numero di volte, viene detto elemento dominante: si possono memorizzare soltanto gli elementi che sono diversi dal dominante. In questo caso si memorizzano soltanto gli elementi (e la loro posizione) che sono diversi da zero. Grazie a questa accortezza, si può guadagnare un grande risparmio di memoria. Secondo alcune stime, il numero medio di collegamenti da una pagina ad un'altra è di 10 link per pagina. Mettendo insieme le cose si vede che, laddove nella matrice di Google G la complessità in spazio richiesta era dell'ordine di $O(n^2)$, nella matrice H è richiesta una quantità di memoria dell'ordine di $O(10n)$, un notevole miglioramento (uno sforzo lineare contro uno sforzo quadratico).

Ritornando all'esempio precedente, questo significa dover memorizzare per la matrice soltanto 8.1E10 elementi double, il che richiede 6.48E11 B, cioè ancora, arrotondando per troncamento, 603 GB.

3.3 Conclusioni

Se si pensa che tra le macchine standard che si vedono in commercio nei punti vendita (dove per standard si intendono quelle macchine non prodotte per scopi particolari) iniziano a comparire modelli con una dotazione di 8 GB di RAM e più, questo significa che anche poche macchine non dedicate possono lavorare con matrici di grandissime dimensioni.

Vi sono altri metodi che possono fare lo stesso calcolo, come GMRES o BICSTAB o altri ancora, ma contrapposto al pregio di essere più veloci, c'è il difetto di richiedere molta più memoria di quella richiesta dal metodo delle potenze, che in definitiva è quello che meglio si applica in questo caso.

Capitolo 4
Cicli e affossamenti

Questo capitolo descrive situazioni in cui possono ottenersi cicli ed affossamenti, presentando tre piccoli casi: in tutti questi casi viene utilizzata la matrice H (non la matrice G) con lo scopo di far vedere cosa succede quando la matrice non ha le proprietà richieste.

Un ultimo esempio mostra una situazione differente in cui è stata invece usata la matrice G (e quindi non sono presenti situazioni anomale): dall'osservazione dei valori finali si possono trarre notevoli considerazioni.

4.1 Un semplice ciclo

Due sole pagine sono sufficienti a creare questo tipo di problema: la pagina P_1 collegata alla pagina P_2 e viceversa, come in figura.

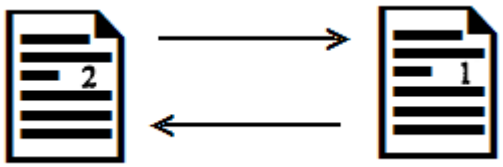

Figura 4.1 – Due pagine che formano un ciclo

La matrice che si ottiene da questa configurazione è la seguente:

$$H = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

mentre il vettore di partenza può essere scelto ricordando il vincolo che la somma delle sue componenti deve essere uguale all'unità, ad esempio

$$p^{0} = (3/4 \quad 1/4).$$

Si ottengono successivamente i seguenti valori:

$$p^{1} = p^{0} H = (3/4 \quad 1/4) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (1/4 \quad 3/4)$$

$$p^{2} = p^{1} H = (1/4 \quad 3/4) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (3/4 \quad 1/4)$$

$$p^{3} = p^{2} H = (3/4 \quad 1/4) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (1/4 \quad 3/4)$$

…

e in generale

$$p^{2k+1} = p^{2k} H = (3/4 \quad 1/4) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (1/4 \quad 3/4)$$

per un numero dispari di iterazioni e

$$p^{2k} = p^{2k-1} H = (1/4 \quad 3/4) \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = (3/4 \quad 1/4)$$

per un numero pari di iterazioni.

Il vettore oscillerà senza mai convergere, non importa quanto grande diventi il numero di iterazioni o quanto tempo passi; la matrice non ha le caratteristiche richieste: in particolare è stocastica ma non primitiva.

4.2 – Affossamenti

Uno schema semplice come il precedente ma con qualche pagina in più è sufficiente per illustrare il caso. Eseguendo un programma che utilizzi il metodo delle potenze con argomenti la matrice che si ottiene dalla figura 4.2 e il vettore iniziale si ottiene la risposta riportata in seguito.

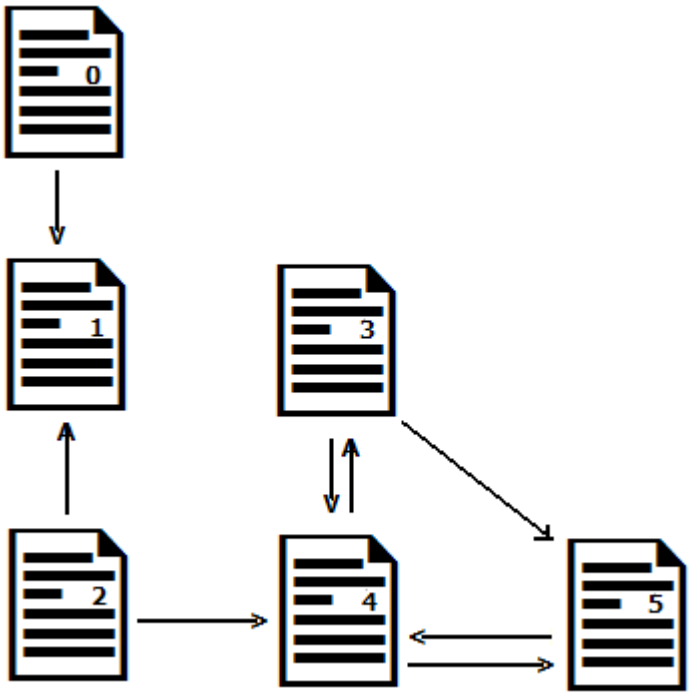

Figura 4.2 – Una configurazione per l'esempio sugli affossamenti

Anche qui si sceglie un vettore iniziale e si ricava la matrice H:

$$p^{0} = (1/6 \quad 1/6 \quad 1/6 \quad 1/6 \quad 1/6 \quad 1/6), \quad H = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

e dopo 25 iterazioni il vettore si stabilizza ai valori:

$$p^{25} (0{,}000000 \quad 0{,}000000 \quad 0{,}000000 \quad 0{,}259259 \quad 0{,}129630 \quad 0{,}194444)$$

dove le ultime tre componenti accumulano tutto il punteggio disponibile (da notare anche che la somma dei punteggi non è uguale all'unità).

4.3 Cicli e affossamenti

L'ultimo caso mostra cosa succede se in una data configurazione coesistono cicli e affossamenti: non solo una parte delle componenti del vettore resteranno nulle, ma le restanti oscilleranno indefinitamente.

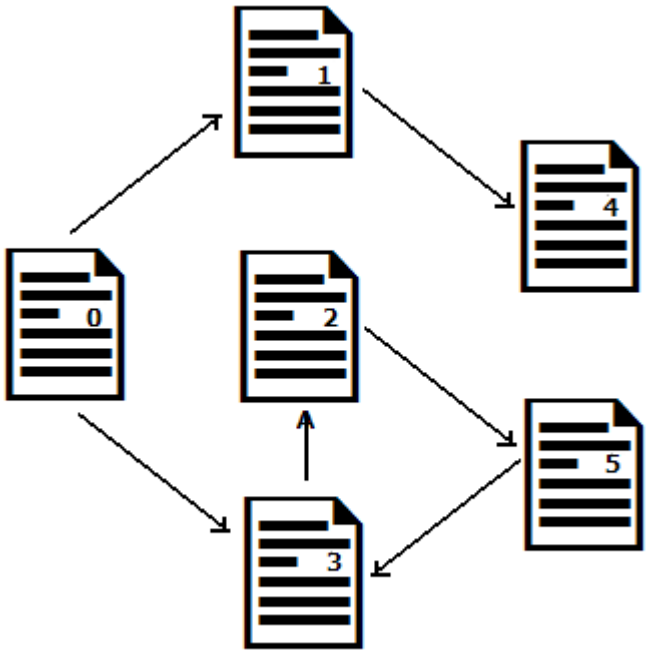

Figura 4.3 – Una configurazione per l'esempio contenente affossamenti e cicli

Di seguito il vettore iniziale, la matrice H e una sequenza di iterazioni che mostra sia le componenti che hanno valore zero, sia le componenti che presentano oscillazioni:

$$p_0 = \begin{pmatrix} 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \end{pmatrix}$$

$$H = \begin{pmatrix} 0 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

p^1 = 0,000000 0,083333 0,166667 0,250000 0,166667 0,166667

p^2 = 0,000000 0,000000 0,250000 0,166667 0,083333 0,166667

p^3 = 0,000000 0,000000 0,166667 0,166667 0,000000 0,250000

p^4 = 0,000000 0,000000 0,166667 0,250000 0,000000 0,166667

p^5 = 0,000000 0,000000 0,250000 0,166667 0,000000 0,166667

...

Questi problemi (cicli e affossamenti) vengono risolti costruendo la matrice G, con le proprietà descritte (ottenendola come somma di due matrici per limitare fortemente la richiesta di risorse utili al calcolo).

Nel quarto ed ultimo esempio si presenteranno solo il vettore iniziale, la matrice e i valori all'iterazione finale, volutamente senza l'immagine; se ne darà l'interpretazione numerica ma si riprenderà l'esempio nel prossimo capitolo, completandolo della figura ed esplicitando i fatti sottili che i numeri (del vettore e della matrice) dicono, ma che non rendono immediatamente comprensibili, facendo vedere

come la progettazione accurata dei siti possa aumentarne la visibilità (questo è molto importante soprattutto per quelle aziende che decidono di condurre le loro attività in rete: se negli affari tradizionali le aziende erano contrassegnate dalle insegne più luminose, visibili a grande distanza ed erano i clienti che si muovevano per dirigersi verso gli edifici ospitanti le aziende stesse, oggi nell'e-commerce le insegne sono spesso sostituite dal PageRank e sono sempre più le aziende a muoversi nelle case dei clienti, potenziali o no, ogni volta che i navigatori "aprono" i loro browser).

In breve, si vedrà che la disposizione degli elementi all'interno dei siti può influenzare il calcolo del PageRank di un sito e quindi il suo "voto" e di conseguenza la sua visibilità.

4.4 Significato dei valori del vettore stazionario

"Vettore stazionario" è il nome con cui si designa il vettore contenente i valori di PageRank delle pagine prese in considerazione.

Supponiamo di ricavare dalla configurazione di un sito i seguenti elementi:

$$p^{0} = \frac{1}{8} e_R$$

$$H + A = \begin{pmatrix} 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/5 & 0 & 0 & 0 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \\ 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \\ 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \\ 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 & 1/8 \end{pmatrix}$$

e il vettore finale:

$$p^{58} = \begin{pmatrix} 0.3030 & 0.1355 & 0.1355 & 0.1355 & 0.0727 & 0.0727 & 0.0727 & 0.0727 \end{pmatrix}$$

L’interpretazione è la seguente: la prima componente indica che il 30,30% del tempo è impiegato dal navigatore per visitare la prima pagina; il 13,55% per le successive tre e il 7,27% per le ultime quattro. Questo significa anche che la pagina rappresentata dalla prima componente è la più importante secondo la definizione di importanza del PageRank di Google. Gli altri valori, indicano il grado di importanza delle altre pagine.

Capitolo 5
Conseguenze notevoli

In questo capitolo si osserva come il PageRank influisce (o come può influire) sulla progettazione di siti web: quali sono gli accorgimenti per ottenere un voto maggiore, cosa è da evitare, cosa è utile e cosa è consigliato

5.1 Guadagnare voti.

Consideriamo un semplice sito con poche pagine, ricavandone vettore iniziale, matrice e vettore finale:

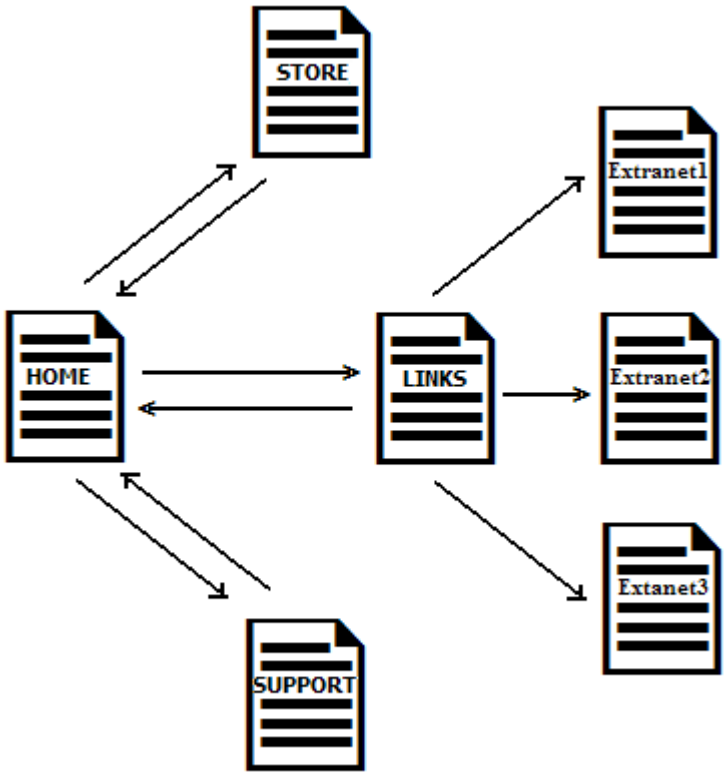

Figura 5.1 – Una semplice configurazione per un sito di e-commerce.

$$p^{0} = \frac{1}{7} e_R$$

dove e_R è il vettore riga con componenti uguali al valore 1 e

$$G = \begin{pmatrix} 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/4 & 0 & 0 & 0 & 1/4 & 1/4 & 1/4 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \end{pmatrix}$$

$$p^{61} = \begin{pmatrix} 0.3252 & 0.1433 & 0.1433 & 0.1433 & 0.0816 & 0.0816 & 0.0816 \end{pmatrix}$$

In questo sito la home ha il PR più alto, in accordo alla politica dei voti (ha un numero maggiore di collegamenti entranti rispetto alle altre pagine); ma allora, se si aumenta il numero dei link entranti alla home, si può ottenere un aumento dell'importanza della stessa.

Si aggiungono così altre tre pagine all'interno del sito: queste tre pagine hanno link entranti provenienti dalla pagina dei link e link uscenti diretti alla home, come nella figura 5.2.

Ci si aspetta, in accordo al calcolo del PageRank, che per la home si verifichi una crescita del valore indicante la sua importanza.

Calcolando il vettore iniziale, la matrice e il vettore per questa configurazione otteniamo:

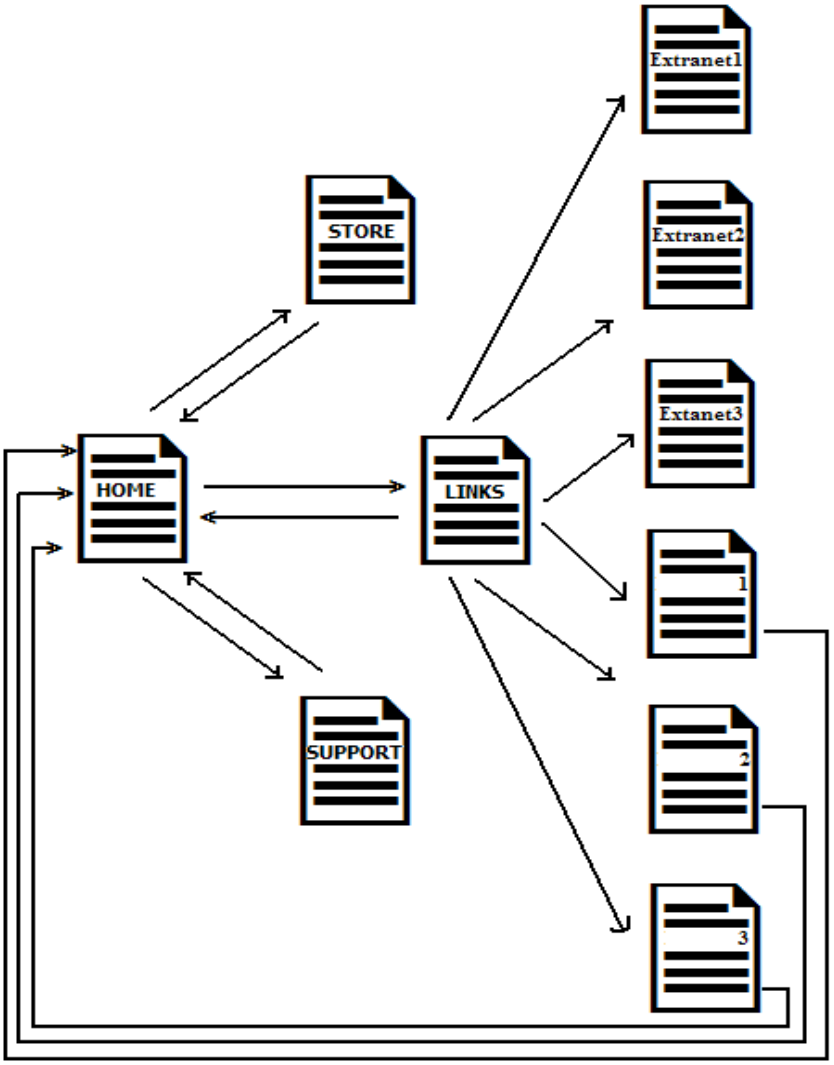

Figura 5.2 – L'aggiunta di alcune pagine al sito della figura precedente

$$p^{0} = \frac{1}{10} e_R, \quad G = \begin{pmatrix} 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/7 & 0 & 0 & 0 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 \\ 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 \\ 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 & 1/10 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

e

$$p^{44} = (0.365 \quad 0.129 \quad 0.129 \quad 0.129 \quad 0.041 \quad 0.041 \quad 0.041 \quad 0.041 \quad 0.041 \quad 0.041)$$

Confrontando i vettori finali nei due casi precedenti, si vede come nel secondo ci sia stato un incremento del posizionamento della home, intervenendo solamente all'interno del sito.

Confrontando anche i vettori iniziali e le matrici dei collegamenti, si vede invece come non siano immediatamente ricavabili i fatti appena evidenziati.

Tuttavia questo modo di aumentare il punteggio delle pagine dei propri siti va bene soltanto aggiungendo poche pagine; questa strategia è tuttavia sconsigliata qualora si pensasse di voler aggiungere, diciamo, un migliaio di pagine (tecnicamente non vi è nessuna difficoltà pratica nel voler gestire automaticamente una tale quantità di pagine, ma nel seguito sarà chiaro il motivo per cui è controindicato adottare l'uso massivo di questa possibilità).

Si possono fare ancora delle osservazioni, perché è possibile migliorare ulteriormente l'importanza della home di questo sito d'esempio.

Prima di tutto si nota che vi sono dei link a dei siti esterni: questi non apportano nessuna quota, così come sono, per incrementare il punteggio della home. Se anche da queste pagine uscissero dei collegamenti diretti alla nostra home, avremmo l'incremento desiderato. Questa situazione ricorre più frequentemente nel caso di siti partner o di aziende che possono comunque trarre profitto l'una dall'altra; in questo caso i siti esterni restituiscono il favore, puntando a loro volta la home (è una applicazione del vecchio adagio "*l'unione fa la forza*", dove le aziende e i web-master si mettono d'accordo per sfruttare questa opportunità di crescita del PR).

Il sito cambia ancora un po' aggiungendo i collegamenti necessari (vedere fig 5.3); per quanto riguarda il vettore stazionario relativo a quest'altra trasformazione otteniamo:

$$p^{26} = (0.415 \quad 0.132 \quad 0.132 \quad 0.132 \quad 0.031 \quad 0.031 \quad 0.031 \quad 0.031 \quad 0.031 \quad 0.031)$$

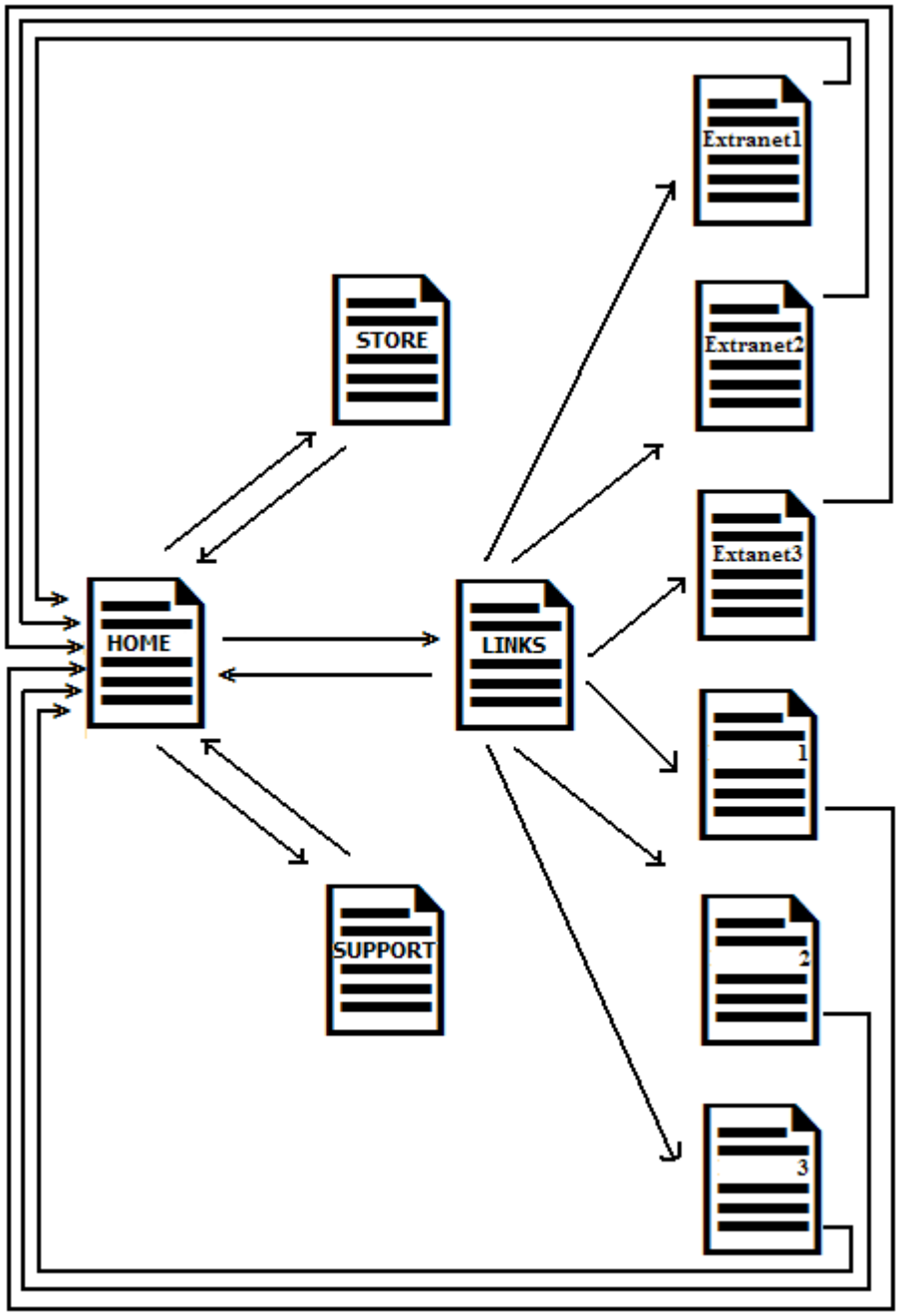

Figura 5.3 – La figura 5.2 a cui sono stati aggiunti alcuni collegamenti

Il punteggio è aumentato ancora. Ma si può fare di più. Come ultimo aspetto, facciamo in modo che il sito sia più semplice e ordinato, eliminando quanto superfluo.

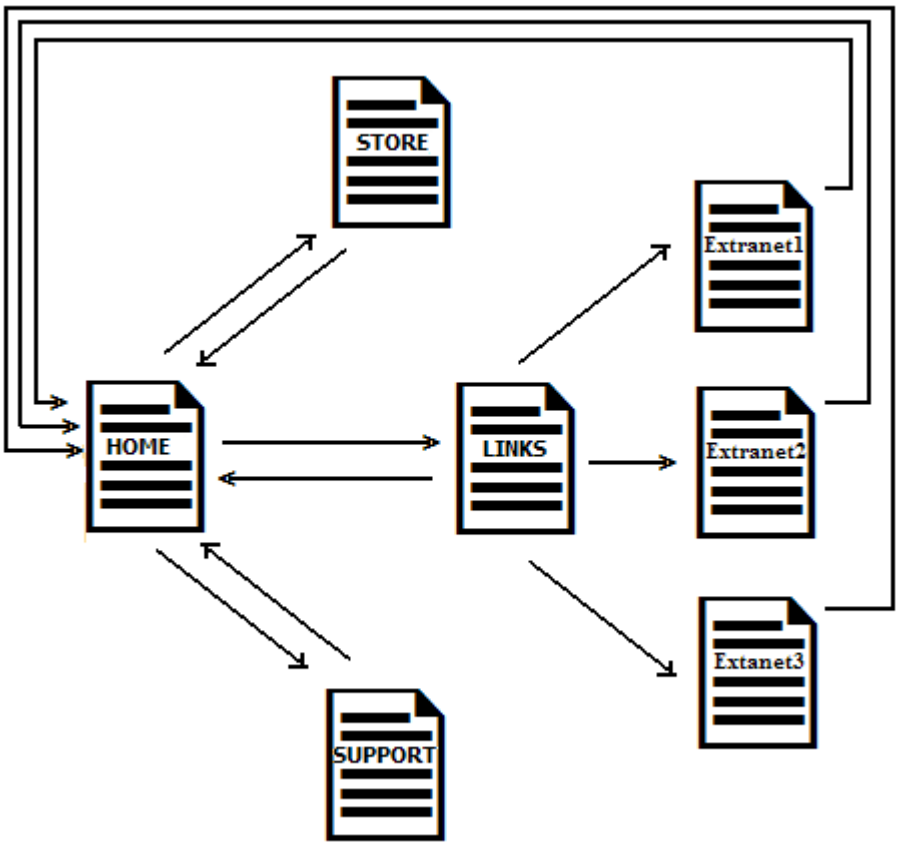

Figura 5.4 – Configurazione ottimizzata al fine del PageRank

Il vettore che otteniamo è il seguente:

$$p^{27} = (0.4224 \quad 0.1411 \quad 0.1411 \quad 0.1411 \quad 0.0514 \quad 0.0514 \quad 0.0514)$$

La conclusione è che un sito ben strutturato può costituire un fattore moltiplicativo "interno" per l'attribuzione dell'importanza da parte dell'algoritmo del PageRank.

L'ultima osservazione che si può fare, mettendo vicini i vettori finali ottenuti, è che una buona progettazione dei siti non solo fa crescere la loro valutazione ma permette a tale valutazione di essere calcolata con un minor numero di iterazioni e, in

definitiva, in un tempo minore (il pedice indica a quale versione del sito si riferisce il vettore):

$$p_1^{61} = (0.3252 \quad 0.1433 \quad 0.1433 \quad 0.1433 \quad 0.0816 \quad 0.0816 \quad 0.0816)$$

$$p_2^{44} = (0.365 \quad 0.129 \quad 0.129 \quad 0.129 \quad 0.041 \quad 0.041 \quad 0.041 \quad 0.041 \quad 0.041 \quad 0.041)$$

$$p_3^{26} = (0.415 \quad 0.132 \quad 0.132 \quad 0.132 \quad 0.031 \quad 0.031 \quad 0.031 \quad 0.031 \quad 0.031 \quad 0.031)$$

$$p_4^{27} = (0.4224 \quad 0.1411 \quad 0.1411 \quad 0.1411 \quad 0.0514 \quad 0.0514 \quad 0.0514)$$

5.2 Situazioni da evitare

Come si è accennato in 5.1, è possibile gestire in modo automatico un gran numero di pagine che puntano tutte alla home, in modo da aumentarne il PR. Tali pagine però sono solamente dei contenitori di link e nient'altro che possa essere utile; in altre parole non portano in esse nessun contenuto. Ma ciò che conta è proprio il contenuto (il contenuto è tutto): inutile avere un libro di pagine bianche.

Queste tecniche possono essere riscontrate presso siti non molto affidabili: il robot di Google è dotato dell'intelligenza sufficiente ad accorgersi di questi casi e il risultato è quello di essere banditi dagli indici.

Un'altra tecnica è quella denominata *PageRank Spoofing*; una pagina con basso PR può essere redirezionata ad una pagina con un grande PR, il che fa acquisire alla pagina a basso PR il PR della pagina più importante (al successivo aggiornamento del PR alla pagina con basso PR sarà assegnato lo stesso PR della pagina con alto PR). Anche queste situazioni possono essere facilmente identificate (se cercando le URL delle pagine sospette il risultato è una pagina totalmente differente allora si ha la conferma che si è in presenza di spoofing).

Ci sono poi casi in cui delle compagnie tentano di vendere link che garantiscono alti PR (questo anche per motivi commerciali); tuttavia anche in questi casi Google ignorerà tali link nel computo del PR.

Un'altra situazione abbastanza comune riguarda quello che si chiama *spamdexing*: è per questo motivo che per l'attributo "rel" di un link HTML è stato

proposto e implementato il valore “nofollow”. Tale attributo è stato fatto per indicare i links che non devono essere presi in considerazione da Google per il calcolo del PR (è il caso degli amministratori di siti di blog ad esempio, dove gli utenti potrebbero produrre molti messaggi contenenti dei link alle proprie pagine: in questo caso gli amministratori possono far in modo che automaticamente venga inserita la regola “rel=nofollow” nei messaggi e, in generale, in tutti i link, risolvendo il problema a monte; tuttavia in questa maniera potrebbero non considerarsi link invece utili).

Una cosa assolutamente da evitare è quella di inserire dei meta-tag (etichette nascoste che servono all’indicizzazione dei motori di ricerca non visibili sulla pagina web) che abbiano nomi uguali a organizzazioni (o , ad esempio, marchi registrati); il sito che usa questa tecnica ha la stessa visibilità di quelle organizzazioni di cui usa il nome: se fatto da un imprenditore questo si configura come illecito concorrenziale (per il quale dispongono costituzione italiana, legge 287/90 per la tutela della concorrenza e del mercato, articolo 2598 c.c. e trattato sull’unione europea artt. 81 e segg.), sanzionabile secondo codice civile (il caso di riferimento è il caso Genertel).

In altri casi (linking e framing), anche se non si provoca una crescita del PR, bisogna ricordare che può essere comunque violato il diritto d’autore.

Le manipolazioni del PR precedenti riducono la qualità e la rilevanza delle ricerche; é per questo che Google elimina dagli indici quei siti che usano queste tecniche.

Capitolo 6
Tagging nel web

Il termine anglosassone “tag” significa etichetta. Mentre “tagging” indica l’attività di etichettatura, cioè l’abitudine di assegnare a delle risorse un termine adatto a classificarle e cercarle.

Ognuno di noi ha dei comportamenti che sono a volte influenzati da usi, tradizioni, credenze, esperienze: non è detto che la visione di qualcosa sia la stessa per tutti e conseguentemente non è detto che la stessa risorsa sia etichettata allo stesso modo da ognuno di noi. Può capitare che gente diversa etichetti la stessa cosa in maniera differente.

Sono molti i siti in cui nascono delle comunità all’interno delle quali i partecipanti interagiscono con le risorse presenti; fra questi siti vi sono quelli che incoraggiano relazioni sociali, che hanno la caratteristica fondamentale di fornire un modo per assegnare tag. La scelta di un tag può essere fatta in maniera formale o informale.

Quando si opta per scegliere un tag in modo formale sono evitati problemi di ambiguità, dando origine a tecniche e applicazioni come quelle che riguardano il web semantico, argomento del primo paragrafo di questo capitolo. Quando il criterio di scelta è informale vengono alla luce tecniche e applicazioni come quelle presenti nel web 2.0, trattato nel secondo paragrafo.

In questo capitolo vengono trattati anche altri ambiti in cui si ricorre al tagging, mentre alle folksonomie, che rappresentano l’incarnazione dell’etichettatura informale delle risorse, viene dedicato per intero il prossimo capitolo.

6.1 Cenni sul Web Semantico

Originariamente il web fu concepito per soddisfare esclusivamente delle richieste: in breve un click su un collegamento provocava la richiesta di un documento da parte del browser al server web, quest'ultimo rispondeva inviando tale documento al browser il quale lo visualizzava all'utente; il tutto avveniva senza una "coscienza" da parte del server o del browser, i quali semplicemente seguivano delle regole, senza "capire" il contenuto del documento richiesto.

Un modo con cui si può attribuire significato al web è mediante l'associazione di meta-dati, informazioni aggiuntive per dare "un senso" alle risorse, che possano descrivere le risorse stesse (insieme ai meta-dati sono necessarie altre accortezze[1], ma per il discorso che si vuole introdurre qui è sufficiente considerare per adesso solo i meta-dati): è possibile ottenere questo risultato perché preventivamente, all'atto della creazione del contenuto, queste informazioni sono state definite e inserite secondo specifiche regole semantiche.

Le meta-informazioni aggiunte hanno un ruolo fondamentale nelle interrogazioni e nelle interpretazioni; ancora oggi i programmi navigano in rete in un mondo di parole per loro prive di significato e per questo motivo nessuno di essi è in grado di controllare e verificare per noi le informazioni che cerchiamo: dobbiamo farlo noi manualmente scartando quelle non volute e approfondendo quelle che ci sembrano interessanti.

Ruolo importante in questo ambito giocano gli strumenti che permettono una rappresentazione della semantica delle risorse. Inoltre è richiesta una particolare cura nella scelta dei meta-dati, attività gravosa per non esperti e questo è un freno allo sviluppo del Web Semantico.

L'utilizzo di questi meta-contenuti costituisce una delle linee evolutive del web che potrebbe veder diminuire la presenza di metodi di ricerca basati su parole chiave e reti di relazioni e connessioni: alle pagine viene aggiunto un significato che ne trascende

[1] Il web semantico deve comporsi di vari livelli fondamentali, necessari ad una semantica ben precisa (vedere 6.1.2)

il contenuto e aiuta i motori di ricerca a reperire solo ciò che si è richiesto (la figura 6.1 ne è un esempio molto semplificato).

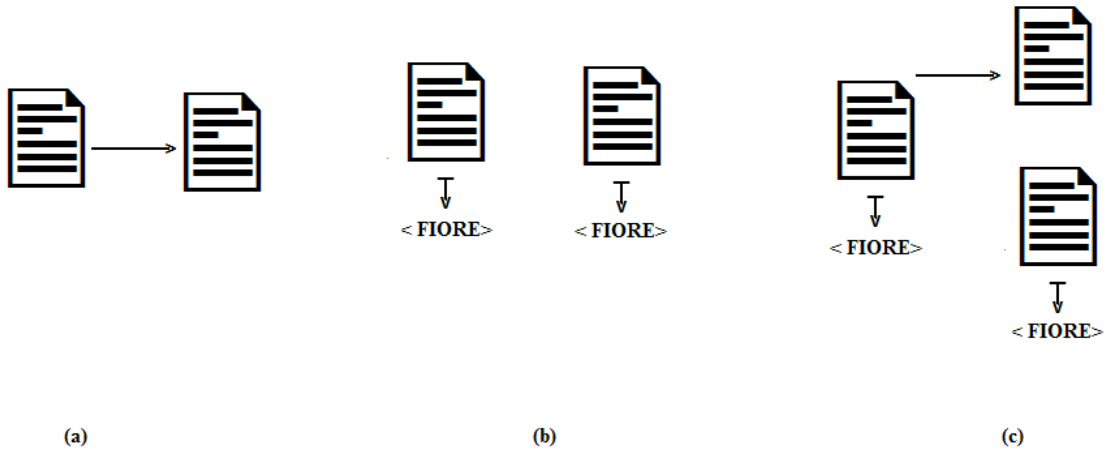

Figura 6.1 – (a) Due pagine con un collegamento ipertestuale (b) Due pagine con un collegamento (non immediatamente visibile) semantico (entrambe hanno contenuti relativi a un qualche tipo di fiore) (c) Tre pagine in cui figurano sia collegamenti ipertestuali che semantici.

In definitiva con il termine Web Semantico si intende "una miglioria" del Web attuale che lo renda meno umanocentrico per tentare di risolvere l'incapacità dei calcolatori di comprendere i contenuti delle risorse (si parla di contenuti "machine processable" o di contenuti "machine interpretable"); lo si vuole rendere più espressivo in modo che i contenuti possano essere interpretati non soltanto da noi ma anche dalle macchine e quindi possano essere svolte automaticamente tutta una serie di attività senza il bisogno di una supervisione umana (una prima conseguenza è una ottimizzazione nella ricerca di informazioni): è la via per una Internet "pensante". Un lettore umano non farebbe fatica a capire l'argomento trattato leggendo la sequenza di tag presentate dal server, ma un agente software sarebbe fortemente limitato.

Benché il web semantico non sia ancora una realtà (se non in piccoli ambiti), molte delle tecnologie nate per comporlo sono disponibili.

6.1.1 – Descrizione sommaria delle tecnologie per il web semantico

Come detto il web semantico è una miglioria del web attuale: nulla viene rimpiazzato ma l'esistente viene arricchito ed esteso con nuove entità disposte su più livelli (fig. 6.2):

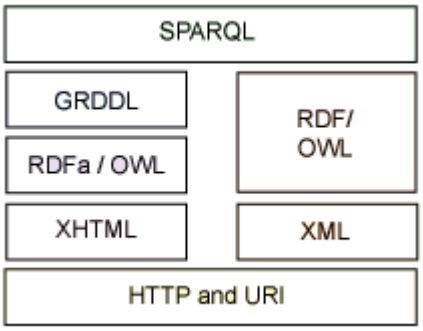

Figura. 6.2 – I livelli del web semantico

Il primo livello è quello che costituisce anche il web "normale"; dato che il web semantico ne è un estensione, include questo livello.

Facendo un paragone con la grammatica:

- Gli URI sono i nomi con cui il web semantico identifica qualunque tipo di risorsa[2];

- HTTP comprende invece i verbi (GET, PUT, POST)

Al livello due compaiono XML e XHTML: permettono di descrivere relazioni; il problema è che esistono diversi modi per farlo, il che può portare ad ambiguità. La soluzione si è trovata nel livello successivo.

Il livello tre comprende diverse componenti:

---

[2] Purtroppo con il divenire dinamico del web nomi significativi come http:/www.mysite.com/offer.html sono divenuti meno semantici, come http://www.mysite.com/inventory.cgi?pid=1234; i sistemi di Content Management Systems e Web Development Framework cercano di risolvere questo problema. Inoltre può succedere di offrire contenuti identici tramite diverse URI (si consideri ad esempio http://www.mysite.com/item.html e http://www.mysite.com/item.php): questo è un altro problema di cui soffre la semantica degli URI (allo stesso modo ne paga le conseguenze il ranking attribuito da Google, perché il motore di ricerca può essere meno preciso). Una maniera elegante per risolvere questa situazione è quella che fa uso degli headers http per richiedere il tipo di contenuto (sotto forma di HTML con la richiesta text/html o sotto un'altra forma con la richiesta application/rdf+xml).

- RDF è un linguaggio derivato dall'XML che permette di descrivere relazioni del tipo soggetto-predicato-oggetto in modo univoco; tuttavia non tutti i produttori rendono i loro contenuti disponibili in questo formato (probabilmente per ragioni di complessità);

- RDFa è la risposta di XHTML ad RDF (altra problematica è la compatibilità fra RDF e RDFa: il secondo non è idoneo per semantiche che richiedono il primo);

- GRDDL risolve il problema del passo precedente: è un metodo che rileva la presenza di RDFa da è capace di estrarre RDF

- OWL: a questo punto entrano in gioco le ontologie e i linguaggi per gestirle; in breve servono per esprimere un vocabolario comune; un semplice esempio per rendere chiara l'utilità è il seguente: si considerino due siti (A e B) che utilizzando l'RDF definiscano le relazioni A:<risorsa><autore><nome> e B:<risorsa><produttore><nome>. Rendendo un discorso complesso molto semplice, le ontologie sono un modo di dire che autore e produttore in questo ambito sono la stessa cosa. Possono anche indicare che due stringhe diverse (perché relative a due lingue diverse) significano la stessa cosa. E' proprio grazie alle ontologie che viene migliorata la capacità di comprensione delle macchine. Non dimentichiamo però che il web è immenso e ricoprirlo tutto con delle ontologie è un compito difficile (bisogna anche tenere conto del fatto che le ontologie hanno significato solo nell'ambiente, o dominio, per il quale sono state progettate e si presenta quindi il problema di come far comunicare tra loro ontologie diverse).

Al livello quattro troviamo l'ultimo pezzo del mosaico:

- SPARQL è un linguaggio di interrogazione simile all'SQL costruito appositamente per interrogare insiemi di dati che sono in formato RDF, dati provengono da diverse fonti. Ora, i dati che sono il risultato della combinazione di altri dati provenienti dalle fonti più disparate, non hanno una struttura ben definita. La flessibilità e la potenza di questo linguaggio di interrogazione sta proprio nel fatto che è più idoneo dell'SQL standard per interrogare tali insiemi di dati.

6.1.2 – Realtà di utilizzo

In alcuni ambiti sono già state sviluppate applicazioni del web semantico, dove è maggiore il bisogno di integrare i dati prendendoli da diverse fonti: produttori di energia, di farmaci, istituzioni governative e biblioteche digitali ne sono alcuni esempi e i produttori di programmi per computer stanno definendo, o lo hanno in parte già fatto, ontologie e regole degli specifici domini di appartenenza.

Esistono portali che offrono la possibilità di fare ricerche più efficaci basate su ontologie come Gopubmed (www.gopubmed.org), JeromeDL (www.jeromdl.org), NASA (sweet.jpl.nasa.gov), Museo Suomi (www.museoswomi.org), Gene ontology (www.geneontoly.org).

Medicina e biologia sono i più attivi a questo proposito: ad esempio sistemi di supporto ai medici basati su tecnologie del web semantico e di intelligenza artificiale sono già in funzione. Una prima versione che mostra le potenzialità del sistema è del 2008 (http://www.heartfaid.org/). Ospite della versione definitiva è il 2009.

Il coordinamento tra i vari gruppi che si occupano di realizzare e proporre standard per il web semantico oltre che per il web è prerogativa del W3C Multimedia Semantics Incubator Group. Il lavoro è quello di fornire esempi d'utilizzo delle tecnologie ed evidenziare insufficienze o mancanze degli attuali strumenti tecnologici, di cercare di rendere i sistemi informatici capaci di scambiarsi e usare mutuamente informazioni codificate utilizzando i differenti standard multimediali esistenti e di mostrare il valore aggiunto del web semantico.

Nella Tabella 6.1 si sono prese in considerazione alcune caratteristiche del web semantico suddivise fra quelle che possono determinarne il successo e quelle che ne frenano la realizzazione; da una parte sono state considerate la struttura dei dati (che permette una loro migliore elaborazione), la ricerca delle informazioni (perché e legata al ranking di cui ci si è occupati in questa tesi) e la forma dei contenuti (perché si vuole che essi possano essere gestiti automaticamente); dall'altra parte sono state considerate la architettura e infrastruttura (quali supporti alla progettazione), la progettazione e la complessità d'uso.

WEB SEMANTICO	
PRO	CONTRO
Dati strutturati.	Architettura e infrastruttura complesse.
Ricerche più veloci e accurate.	Progettazione difficile (delle ontologie).
Contenuti comprensibili ad umani e pc.	
Semplicità d'uso	

Tabella 6.1 – Punti forti e punti deboli del web semantico

6.2 Web 2.0

Sostanzialmente si tratta di un nuovo modo di usare il web, sempre più orientato alle persone e all'interazione sociale. Le informazioni possono circolare liberamente fra i siti, il più delle volte con modalità e scopi diversi da quelli previsti dal loro produttore. Questo è possibile per la stessa natura del Web 2.0 che è una entità pensata con lo scopo di condividere e quindi non si tratta di approfittare del lavoro altrui (questo però non vuol dire che non si debba tenere conto dei temi relativi alla proprietà intellettuale e alla proprietà delle informazioni). Contemporaneamente sono offerte opportunità di lavoro e di informazioni impensabili prima e che possono crescere su strati di informazioni precedentemente acquisite nei siti più disparati. E' la metamorfosi della rete da un insieme di semplici siti ad una struttura in cui le varie componenti sono in grado di interagire ed elaborare in maniera collettiva. Il web assomiglia sempre più ad una grande applicazione dove si abbandona la classica navigazione sequenziale fra pagine statiche. Si riesce a valorizzare i contenuti in modi nuovi e creativi e i servizi e le tecnologie basano il loro successo in funzione del numero di altri consumatori che li offrono o li utilizzano.

Nel web 2.0 è normale ricercare, filtrare e *remixare* diversi tipi di contenuto in nuovi oggetti di informazione. E' proprio in questo che risiede il grande potenziale del web 2.0. Internet non è più una "semplice" rete delle reti, ma un luogo abitato e vissuto

quotidianamente in cui gli abitanti sono in costanti rapporti tra di loro e hanno assunto un ruolo primario nella produzione di contenuti divenendo fruitori attivi. Mentre prima erano solo i computer ad essere connessi tra loro, adesso sono collegati sia milioni di computer sia milioni di persone che instaurano relazioni e collaborazioni e dove l'imperativo è "aggregare contenuti".

Nella metamorfosi è inclusa la trasformazione in piattaforma: non è più necessario installare software nella propria macchina e lo stesso software viene distribuito e aggiornato in maniere diverse dalle consuete: quando l'utente si connette è il browser stesso, se ve ne è bisogno, che richiede l'autorizzazione per l'aggiornamento e provvede a scaricare, installare e utilizzare l'ultima versione della porzione di codice necessaria (per la maggior parte codice indispensabile per gestire l'interfaccia e l'interazione col sistema). Si possono utilizzare applicazioni la cui elaborazione avviene tramite la cooperazione tra browser e server. Si costruiscono applicazioni su misura acquisendo ciò di cui si necessita da fonti diverse componendo il tutto al bisogno (questo amalgama di web application è conosciuto con il nome di *mash-up*).

La nuova piattaforma web considerata dal punto di vista aziendale significa integrazione di infrastrutture e servizi (IT + SOA): la visione è quella dell'interoperabilità fra sistemi non omogenei, sviluppati da fornitori differenti o fra sistemi datati e nuovi.

Di fondamentale importanza e caratteristica principale è la partecipazione degli utenti (sia aziende che privati) e il loro pieno coinvolgimento. Gli utenti Internet sono sempre maggiormente rappresentati dai propri contributi. Il successo di questi siti e la loro conseguente proliferazione non può essere giustificata solamente da uno spirito di condivisione. Gli utenti hanno degli scopi ben precisi che tentano di perseguire sfruttando al massimo le diverse funzionalità per dimostrare le proprie capacità commentando i contributi di altri per poi ricevere feedback positivi sui propri contenuti. L'utilizzo di strumenti Web 2.0 evidenzia la necessità dei singoli di autopromuoversi attraverso le opportunità date dalla disponibilità di nuovi strumenti.

Infine bisogna sottolineare ancora una volta che il termine web 2.0 non indica una nuova versione. Il numero 2.0 non indica un aggiornamento del web, come avviene per i

software; è quindi un uso un po' improprio con cui si intendono descrivere alcune caratteristiche del web di oggi. Nel caso del web 2.0 si vuole enfatizzare un nuovo modo di utilizzare gli strumenti e le tecnologie esistenti, sia da parte di progettisti che da parte di utenti, con l'obiettivo di aiutare la creatività e la condivisione di informazioni (una conseguenza importante è costituita dall'insieme di interazioni che possono avvenire tra gli utenti). Si deve tener conto di tutta quella serie di fattori evolutivi nel passato del web che lo hanno portato al punto in cui si trova, in maniera velocissima per quello che può vedere un essere umano, ma in maniera lentissima per quello che può essere considerato il tempo visto da una CPU. Ciò che ha portato il Web alla sua forma attuale è anche merito degli Web Standard, sempre più accettati e adottati da parte dei browser (non dimentichiamo le incompatibilità e le incoerenze fra i browser che nel passato erano superiori in numero rispetto alla situazione attuale). Ancora, l'espansione dell'insieme di tecnologie server side accolte e una sempre più ampia cura verso l'usabilità e l'accessibilità.

In conclusione il Web 2.0 è il riflesso di una nuova concezione del design. Gli strumenti erano già disponibili da parecchio tempo ma soltanto da poco li si utilizza in modo appropriato. Si potrà intendere il termine Web 2.0 come una "nuova versione" del Web solo quando saremo di fronte a tecnologie e infrastrutture completamente nuove (se si considera la struttura portante ciò che regge la rete è sempre l'insieme dei protocolli di comunicazione degli anni '60-'70).

6.2.1 – Tecnologia del web 2.0

Fra le tecnologie che costituiscono il web 2.0 notiamo AJAX, un approccio di sviluppo *web* basato su JavaScript ed XML. Sono tecnologie usate dal client con le quali vengono prodotte interfacce sempre più innovative ed efficienti (molte operazioni fanno a meno del server). Inoltre vengono scritti quelli definiti plugin che permettono di interagire con i siti web. Dal lato server vengono invece utilizzate PHP, JSP, ASP il cui risultato sono forum, wiki, blog,… E' questo miscuglio che fa funzionare il web come una grande applicazione.

Un'altra tecnologia usata è RSS che permette di ottenere aggiornamenti automatici non appena un sito cambia, anziché controllarlo ogni volta per avere le ultime

informazioni. E' sufficiente iscriversi al *feed* RSS del sito e non appena il contenuto di tale sito cambia, viene automaticamente inviato al lettore o aggregatore di RSS.

Le open API permettono la comunicazione fra programmi diversi altrimenti incompatibili: l'accesso ad ampi database informativi è assicurato e ancora una volta questo significa riutilizzo per creare nuovi *mix* e combinazioni.

6.2.2 – Caratteristiche del web 2.0

Nella seguente tabella, considerazioni analoghe a quelle fatte per il web semantico prendendo in considerazione le stesse caratteristiche

WEB 2.0	
PRO	CONTRO
Dati non strutturati.	Contenuti comprensibili solo da umani.
Ricerche veloci.	
Architettura, infrastruttura, progettazione: semplicità d'uso	

Tabella 6.2 – Punti forti e deboli del web 2.0

Facendo un paragone con la tabella 6.1 si vede come il web 2.0 sia avvantaggiato rispetto al web semantico. Tuttavia, data la sempre crescente richiesta di automazione in tutti i campi, è possibile che, trascorso un certo intervallo di tempo, il web semantico si espanda almeno quanto il web 2.0. Anzi, vi sono delle indicazioni che fanno pensare che i due si fonderanno in una unica entità, perché è sempre più evidente la simbiosi fra le esigenze di un calcolo veloce e la capacità umana di comprendere le cose. In un prossimo futuro magari sarà possibile interagire con un agente intelligente in linguaggio naturale.

6.2.3 – Esempi

Una utile applicazione è la realizzazione di SI per la PA, la reingegnerizzazione dei processi interni delle PA e l'aumento della qualità dei servizi offerti al cittadino.

Altre applicazioni di una certa importanza sono le attività di e-learning che possono essere sfruttate (si veda più avanti dove si accenna ai blog)

6.3 – Web semantico vs Web 2.0

Un interessante strumento datoci da Google (Google Trend), ci può fornire il numero di richieste di ricerca servite riguardo una o più parole di interesse. Chiedendo di comparare le richieste circa i termini "Web Semantico" e "Web 2.0" si nota una differenza enorme fra i due a sfavore del Web Semantico; questi risultati possono servire come misura indiretta del fatto che il Web Semantico non è di interesse oppure che non è affatto conosciuto oppure ancora che è scarsamente sviluppato (fig 6.3). In particolare si può vedere un certo divario digitale (o culturale) fra gli Stati Uniti e l'Italia dove il numero di richieste relative al termine Web Semantico (in Italia) viene addirittura trascurato essendo molto piccolo rispetto alle richieste relative al termine Web 2.0.

Generalmente le cose, astratte o no, che sono rivolte a un maggior numero di persone, decretano il loro successo, rispetto a quelle che sono rivolte a una limitata cerchia di soggetti. Personalmente ritengo che questo sia uno dei motivi per cui il Web Semantico riscuote poco successo (sono ancora pochi i siti che applicano le tecnologie relative). Probabilmente nei prossimi anni il dominio sarà ancora segnato dal Web 2.0 e dalle tecnologie che si evolvono in quella direzione, ma con il procedere dei progressi tecnologici e con l'ingresso in scena di agenti software sempre più numerosi per i quali è necessaria una semantica, il Web 2.0 stesso si evolverà nella direzione del Web Semantico e i due si fonderanno in un'unica entità, magari supportata da una nuova tecnologia e nuovi protocolli in sostituzione degli attuali. Questioni di tempo.

Figura 6.3 – Confronto fra Italia e Stati Uniti relativo al numero di richieste di ricerca fatte e Google dei termini Web 2.0 e Web Semantico. Come si vede, in Italia il Web Semantico è ancora praticamente sconosciuto.

6.4 – Altri ambiti in cui è utilizzato il tagging

Il tagging non si riscontra solo nelle applicazioni reali riguardanti i due casi precedenti, ma anche in altri casi. Fra i tanti, alcuni sono riportati in seguito. Interessanti sono le applicazioni didattiche che possono essere messe a disposizione.

6.4.1 – blog

Sostanzialmente è un sito in cui l'autore (o un gruppo di autori) pubblica degli articoli, "aiutato" da guide automatiche per la costruzione, personalizzazione e visualizzazione di pagine in cui gli articoli sono ordinati su base cronologica e sono chiamati post. I lettori possono aggiungere dei contenuti (commenti). Ruolo chiave gioca la facilità con cui si possono commentare gli articoli, il che può creare poi legami fra autori di blog diversi (ed ecco un primo esempio di rete sociale). I blog non sono poca cosa se si pensa che Technorati, motore di ricerca specializzato per i blog, ne conta più di 83 milioni, continuamente aggiornati e mantenuti.

Il fatto poi che vi sia una così grande libertà di espressione ha portato a stabilire delle regole di sana convivenza civile (ad esempio nel caso di disaccordi si suggerisce di usare i canali privati invece di quelli pubblici; si consiglia anche di ignorare quegli utenti che utilizzano gli strumenti in maniera sgradevole; non per ultima ci si aspetta che gli utenti si assumano la responsabilità di ciò che dicono).

Esempi di blog fra quelli maggiormente discussi in questo periodo sono quello di Beppe Grillo o di Antonio di Pietro o della BBC per i suoi giornalisti; e ancora quello, per portare un esempio di uso negativo, dell'egiziano Suliman che nel 2006 diffamò il Presidente della Repubblica e incitò all'odio dell'Islam.

Gli uomini d'affari e le aziende stesse non sono rimasti indifferenti al fenomeno e hanno trovato nei blog aziendali e pubblicitari uno strumento utile ad arricchire la loro capacità comunicativa con i clienti e con i consumatori (anche qui purtroppo alle volte si sono registrate tecniche di spamming; ne abbiamo accennato parlando delle tecniche per aumentare il PageRank e in questo caso il termine utilizzato è "Splog"). Vi sono anche dei mezzi che esaminano i contenuti prima della pubblicazione per evitare casi di spamming: Akismet è per esempio una società che offre questi servizi mentre in altri

contesti si utilizzano dei test per escludere i sistemi automatici. Il classico esempio è la richiesta di compilare dei campi inserendo un numero che si vede nella pagina: i bot non sono in grado – almeno non ancora – di riconoscere un numero da un disegno.

Una accusa rivolta ai blog è che soltanto alcuni ne traggono veramente profitto, cioè, in particolare, chi detiene le tecnologie che, grazie alla produttività di tanti, riescono ad ottenere popolarità e anche introiti notevoli.

6.4.1.1 – Esempi educativi di blog

Il primo esempio è quello di un gruppo di utenti (in gergo "bloggers") che utilizzano i loro blog individuali ma in modo di "unirli" per formare conoscenza tramite l'interconnessione di post e commenti; questo gruppo di utenti può essere costituito da allievi di una classe o istituto sotto la guida di un docente. I docenti possono servirsi del blog per dare notizie riguardanti i corsi e per avere un feedback dagli studenti. Si possono poi integrare tecnologie di aggregazione delle notizie per permettere ai gruppi di essere aggiornati nel momento in cui compare un nuovo post. Questo esempio ricorda un po' l'utilizzo di Moodle che si fa nella nostra facoltà (e a questo proposito si potrebbe pensare di estenderne le funzionalità).

Il secondo esempio è l'Università di Warwick (http://blogs.warwick.ac.uk/) che per prima, nell'ottobre del 2004, ha garantito servizi web a livello istituzionale ai propri studenti. Gli scopi erano molteplici: osservare a cosa poteva portare l'adozione dei blog, promuovere la comunità in cui l'educazione era vista come funzione secondaria e sviluppare un sistema che, a differenza degli altri sistemi commerciali presenti, fosse in grado di far fronte alle esigenze dell'università, in particolare permettendo l'integrazione con altri sistemi universitari. Il sistema è molto usato e conta 4.540 blog, 13.255 tag, 88.619 accessi, 111.803 immagini e 190.859 commenti.

6.4.2 – wiki

Sono siti web in cui ogni utilizzatore può modificare o aggiungere nuovo contenuto ad un insieme di pagine interrelate, utilizzando un processo di creazione ed editing. Anche qui la fa da padrona la semplicità. A parte i soliti WikiGremlin o "guastafeste",

ai quali si cerca di impedire la possibilità di intervenire, il sito è di tipo collaborativo e si può scrivere ciò che si vuole. Il limite è stabilito da un certo controllo da parte degli altri utenti, che garantiscono in una certa misura la veridicità dei contenuti. Ma il fatto che vi sia questa supervisione non esclude la possibilità che vi sia del materiale incerto o non privo di difetti ed errori (manomissioni e scherzi di cattivo gusto)

L'utilità di questo tipo di siti è molto chiara se si pensa all'esempio numero uno: WikiPedia, di fatto la più completa e affidabile enciclopedia libera del web. Ruolo fondamentale gioca l'automoderazione della comunità, grazie alla quale è difficile trovare informazioni false o tendenziose.

Anche i wiki sono stati criticati, in particolar modo Wikipedia stessa la quale, secondo alcuni sarebbe "al servizio" di grandi multinazionali, esprimendone il punto di vista e non essendo quindi imparziale o libera come si vuole fare credere.

6.4.2.1 – Esempi educativi di wiki

L'esempio riguarda l'utilizzo allo scopo di creare progetti di classe, attività molto utile per l'accumulazione in maniera incrementale del sapere da parte di un gruppo di produzione (in uno dei corsi della nostra facoltà era richiesta la partecipazione ad almeno un wiki). I docenti possono fornire un input iniziale per dare una forma alla struttura che dovrà avere il contenuto e richiedere poi un feedback sui contenuti generati dagli allievi. Gli studenti possono inserire tag su delle porzioni del wiki che meritano attenzione e possono avere dei feedback da parte degli altri studenti.

6.4.3 – Bookmarking Sociale

Servizio offerto agli utenti di un sito in maniera che sia possibile per loro contrassegnare delle pagine ed etichettarne le parti più significative che le descrivono; esempi sono del.icio.us (http://del.icio.us/) e Bibsonomy (http://www.bibsonomy.org/). Con il passare del tempo possono crearsi degli insiemi in cui alle pagine descritte corrispondono gli stessi tag e in questa maniera gli utenti possono ricercare altre voci da contrassegnare con possibili tag. Inoltre, il fatto di poter contrassegnare determinate voci e classificarle con uno o più tag, rende le ricerche all'interno del più efficaci di quelle che farebbero i motori di ricerca su internet. E possibile anche scoprire quali sono

gli altri utenti che utilizzano gli stessi tag e che potenzialmente sono interessati agli stessi argomenti (in questi casi è possibile aggiungerli alla propria rete in modo da essere aggiornati sulle loro attività di tagging relativamente alla voce di interesse comune).

6.4.3.1 – Esempi di utilizzo a scopo didattico

Insegnanti e studenti hanno la possibilità di costruire degli insiemi di risorse anche senza avere tanta esperienza. E da questi insiemi ricavare delle liste di lettura. Utilizzando tag multipli è possibile una strutturazione in sotto-categorie. Voci di interesse comune danno vita a gruppi con interesse comune e gruppi di progettazione.

6.4.4 – Media-Sharing

Scopo è l'archiviazione di media a cui contribuiscono gli utenti a cui permettono di cercare e visualizzare i contenuti. Possono costitutire un supporto per il lavoro creativo e per impegni didattici. Tuttavia il più delle volte vengono utilizzati a scopi ricreativi e i contenuti sono privi di interesse.

Tra gli esempi più noti vi sono YouTube per i filmati (http://www.youtube.com), iTunes per podcast e vidcast (http://www.apple.com/itunes/), Flickr per le foto (http://www.flickr.com/), Slideshare per le presentazioni (http://www.slideshare.net), DeviantArt per il lavoro artistico (http://www.deviantart.com/) e Scribd per i documenti (http://www.scribd.com/). Scribd è particolarmente interessante perché permette di immettere documenti in differenti formati ed è possibile scegliere diversi formati per il download.

6.4.4.1 – Uso didattico

Un buon utilizzo dei podcast sarebbe quello di fornire del materiale introduttivo alle lezioni oppure più in generale fornire le lezioni stesse per facilitare lo studio degli studenti che non possono frequentare. Altra buona applicazione relativa al podcast è quella di fornire materiale didattico linguistico (esempi parlanti di madrelingua). Analogamente per i vidcast: video educativi e registrazioni di seminari possono essere archiviati e resi pronti e disponibili per la condivisione.

Nel caso di flickr si possono annotare, associare e commentare diverse aree di una immagine: questo può facilitare il lavoro dell'insegnante nelle spiegazioni. Una interessante funzionalità è data da flickrCC per il reperimento di immagini rilasciate con licenza CreativeCommons: sono liberamente riutilizzabili come risorse didattiche.

6.4.5 – Sistemi per reti sociali

Permettono di creare delle reti sociali per diversi fini. Esempi sono Facebook (http://www.facebook.com/) e MySpace (http://www.myspace.com) per socializzare, linkedin (http://www.linkedin.com/) per il network sociale professionale, l'ormai noto SecondLife (http://www.secondlife.com/) un mondo virtuale che non è utilizzato solo allo scopo di relax o di divertimento ma per esempio, come è avvenuto recentemente, per simulare una strategia di impresa da adottare poi nel mondo reale a seconda dei risultati ottenuti nel mondo virtuale, Elgg (http://elgg.net/) per incrementare sapere e apprendimento.

Nei sistemi di reti sociali la caratteristica principale è il fatto che l'utente, in qualche modo, descrive se stesso e i propri interessi e generalmente vengono attivate reti di amicizie e comunità. Da queste reti è possibile spostarsi da amici ad amici degli amici e navigare in un modo diverso dalla navigazione tradizionale o semantica: "navigazione sociale", arrivando a percorrere parecchie reti sociali diverse.

I contributi degli utenti vanno di pari passo con le loro reputazioni: si può fare un discorso di ranking in quanto i membri possono essere valutati dalla collettività in maniera differente e quindi alcuni possono assumere posizioni più elevate rispetto ad altri. Si tratta di una piccola, ma sana, competizione che porta i membri della collettività ad essere maggiormente contributivi e di questo ne gode la prosperità della comunità stessa, i cui interessi sono alimentati e sviluppati con costanza.

6.4.5.1 – Uso didattico

Elgg viene utilizzato nell'università di Brighton, molti esperimenti educativi sono supportati da SecondLife (studenti del GoldSmith College hanno realizzato il proprio spettacolo di fine anno su questa piattaforma). Altri sistemi di questo genere sono usati per l'apprendimento professionale all'interno di comunità.

6.5 – Conclusioni

Entrare nel mondo del “Web Sociale” (Semantico, 2.0. blog, wiki,…) è più facile di quel che possa sembrare a prima vista e in realtà la fruizione dei servizi offerti è molto semplice. Scambi di informazioni, servizi di news, aggiornamenti costanti, creazione di contenuti web e servizi, sono tecnicamente realizzabili percorrendo strade diverse.

L’utilità degli esempi considerati è indubbia, sia per chi vuole pubblicare contenuti sia per chi vuole fruirne (a patto che le tecnologie vengano usate diligentemente).

Capitolo 7
Folksonomie

In questo capitolo si focalizza l'attenzione sulle folksonomie, strumenti utilizzati nel contesto del Social Tagging di cui al capitolo 6. Si parlerà di information retrieval in presenza di dati strutturati come le folksonomie

7.1 Introduzione alle folksonomie

Le folksonomie possono essere definite in maniera semplice come delle descrizioni (di informazioni e oggetti) fatte da utenti i quali contrassegnano, con un proprio vocabolario e ad uso personale, informazioni e oggetti in modo da poterli ritrovare al bisogno. Non si dimentichi però che gli utenti appartengono ad un contesto sociale, per cui anche se le descrizioni fatte sono ad uso personale l'utilità di queste descrizioni è estesa a tutta la comunità .

Da un'altra angolazione le folksonomie sono delle attività in cui un certo numero di persone classificano e categorizzano un insieme di oggetti di cui non necessariamente sono esperti. In questo contesto le folksonomie devono il loro nome, in parte, alle tassonomie, anch'esse attività di classificazione e categorizzazione in cui però le persone coinvolte hanno profonda conoscenza degli oggetti. Il termine folksonomia deriva dalle parole folk (gente) e taxonomy (tassonomia) a indicare esattamente quanto precede.

Il bisogno delle folksonomie è evidente solo quando si considerano le dimensioni dell'insieme di oggetti da descrivere: la strada delle tassonomie è percorribile solo in presenza di un modesto numero di oggetti, non certo in contesti quali il web. E' in questi ultimi anni che si comincia a superare la rigidità delle tassonomie tradizionali, spesso inadeguate a rappresentare realtà dinamiche, grazie ai meccanismi di classificazione costruita dal basso forniti dalle folksonomie. Questo fatto può essere preso come un esempio di quello che significa "intelligenza collettiva", dove un certo numero (detto "massa critica") di individui, agendo da filtro, determina quali concetti utilizzare e quali no nella descrizione di un insieme di oggetti.

7.2 Folksonomie e Information Retrieval

Una grande quantità di informazioni richiede sempre un modo per poterle individuare e recuperare; di questo si occupano gli **IRS** (*Information Retrieval System*), che in ambito Internet sono applicazioni usate per la ricerca di risorse (o di informazioni all'interno delle risorse o di metadati che descrivono le risorse stesse), attingendo da basi di dati autonome o distribuite (in questo senso il *World Wide Web* è indicato come una base di dati ipertestuale distribuita), permettendo l'accesso a libri, giornali e altri documenti.

Le folksonomie sono a volte definite come "sistemi di classificazione distribuiti" (Marieke Guy, 2006), creato da un insieme di utenti i quali possono etichettarne le risorse. Anche il ranking è tratto dai contributi dell'utente, sebbene vi siano importanti differenze fra i siti che usano le folksonomie di cui bisogna tener conto, perché differente può essere il modo di gestire le sorgenti nei differenti siti: ad esempio in alcuni siti gli utenti possono attribuire un titolo ai documenti, votarli e commentarli, ma non etichettarli. Dato che il FolkRank è un programma che associa un indice di rilevanza alle risorse di una folksonomia e dato che la rilevanza va di pari passo con i risultati forniti da una richiesta di ricerca è interessante vedere se la ricerca all'interno delle folksonomie è comparabile con la ricerca nel web.

Per questo, dal momento che strettamente legato alla ricerca di informazioni è il tema della loro strutturazione, nel prossimo paragrafo ci occuperemo delle differenze presenti fra alcune forme in cui le informazioni possono essere strutturate: predecentemente si è parlato di web semantico, web 2.0 e di altri sistemi di bookmarking sociale; in 7.3 si confrontano tassonomie, ontologie e folksonomie.

7.3 Folksonomie, Tassonomie e Ontologie

Vi sono parecchi metodi per l'indicizzazione di informazioni con lo scopo di successive ricerche, come ad esempio analisi lessicale, analisi del linguaggio naturale, tassonomie ed ontologie che portano a un indicizzazione sistematica; il problema è che

nessuno di questi metodi supporta l'indicizzazione che può derivare da informazioni che sono generate e collezionate come quelle che provengono dalle folksonomie.

Le etichette possono essere scelte liberamente da ciascuno (per questo sarebbe indicato un ottimo sistema di suggerimenti per l'attribuzione di tag): questo fatto si esprime dicendo che lo "spazio dei nomi" associato alle folksonomie è uno spazio piatto (non gerarchico) in cui non compaiono relazioni come quelle che invece popolano le tassonomie o le ontologie e inoltre non vi sono regole che controllino l'attribuzione di etichette alle risorse.

7.3.1 Confronto e complementarietà

Le folksonomie sono meno costose in termini di sforzo per costruirle rispetto a tassonomie e ontologie. Così pure per quanto riguarda l'aggiornamento e la manutenzione. Altro aspetto di cui tener conto è senz'altro la dinamicità. La seguente tabella mette a confronto folksonomie con tassonomie e ontologie:

FOLKNOMIE	TASSONOMIE E ONTOLOGIE
Costruzione dal basso (definite da utenti)	Costruzione dall'alto (a cura di esperti)
Flessibilità e dinamicità	Forma ben definita: rigidità e staticità
Costruzione veloce e facile	Costruzione onerosa in tempo e sforzo
Cambiamenti immediatamente disponibili	Cambiamenti lenti
Serendipità possibile	Ricerca limitata da classificazione
Orientate a singoli e a comunità	Gestione centralizzata
Nessuna interoperabilità	Interoperabilità

Tabella 7.1 – Folksonomie e tassonomie/ontologie a confronto

7.3.2 Vantaggi e svantaggi

Volendo trarre i vantaggi di folksonomie, tassonomie e ontologie bisognerebbe cercare in qualche modo di unirle.

Per esempio l'unione delle prime due voci della tabella, cioè costruzione dal basso, flessibilità e dinamicità da una parte e costruzione e rigidità nelle regole dall'altra, permetterebbe di poter ottenere la precisione di tassonomie e ontologie in ambienti vasti come il web. Continuando con altre caratteristiche, si avrebbe una costruzione veloce di una struttura aggiornata in tempo reale, il tutto con poco sforzo. Per non parlare poi del vantaggio apportato dalle ontologie che renderebbero disponibile un vocabolario comune tra sistemi differenti che agevoli la comunicazione sia tra agenti software sia tra agenti esseri umani e agenti software.

Altro importante problema è il modo caotico in cui gli utenti etichettano le risorse, non dimenticando gli errori di digitazione, i sinonimi, i plurali. Questi sono alcuni dei motivi per cui sembra sensato che l'unione fra folksonomie, tassonomie e ontologie possa portare buoni frutti, il tutto cercando di fare in modo che nessuno dei tre sistemi di indicizzazione prevalga.

Fra i vantaggi:

1. L'esplorazione per tag fa aumentare la possibilità di trovare nuove risorse rilevanti per la ricerca in corso;

2. Il vocabolario che è utilizzato è lo stesso che viene costruito dagli utenti, piuttosto che da esperti, il che accresce l'usabilità visto che i contenuti sono intellettivamente accessibili al maggior numero di utenti;

3. La terminologia è centrata sull'utente il che meglio riflette il comportamento e i desideri dell'utente stesso;

4. Non richiedendo particolari abilità, non presenta barriere all'utenza, come può invece avvenire con ontologie e tassonomie, appannaggio di esperti;

5. Dispendio economico, di tempo e di energia sono minori che nel caso delle tassonomie e delle ontologie;

6. Mediante la similarità delle etichette permette di scoprire le persone con interessi simili;

7. E' una realtà altamente dinamica e i cambiamenti che avvengono sono immediati e riflettono sempre gli interessi dell'utenza;

8. Allo stesso tempo, permettono sia l'organizzazione delle informazioni di un singolo utente, sia la condivisione con altri utenti e quindi facilitano la crescita come individui e come comunità.

Fra gli svantaggi si possono menzionare:

1. Il grosso problema nell'ambiguità che si può presentare con l'uso di etichette e senza controllo;

2. Altro problema è quello derivante dai sinonimi: la stessa risorsa può essere etichettata in maniere diverse e utilizzando nella ricerca un sinonimo che non compare fra le etichette usate può trascurare elementi importanti che altrimenti verrebbero presentati fra i risultati;

3. Non facendo uso di nessuna gerarchia, funzionano meglio per la ricerca e la scoperta solo nell'ambito di sistemi in cui viene utilizzato il tagging.

4. Nessuna interoperabilità fra sistemi.

7.4 Modelli per folksonomie

Vi sono diversi modelli di rappresentazione delle folksonomie (Hwang, 2007). Eccone alcuni.

7.4.1 Modello a Relazione

L'attività di etichettatura, considerata da un punto di vista del singolo utente, è una semplice relazione T(oggetto, etichetta) in cui l'utente associa una descrizione (mediante etichetta) ad un oggetto.

Quando però l'attenzione si sposta verso un gruppo di utenti, bisogna distinguere anche il singolo, per cui nella relazione si deve tener conto anche di chi etichetta e la relazione diventa T(oggetto, etichetta, utente).

Facendo ancora un passo avanti e notando che esistono diversi "sistemi" che usano le folksonomie (ad esempio del.icio.us, flickr, …) è possibile che uno stesso concetto sia presente in diversi sistemi. Si introduce quindi un'altra informazione nella relazione T che diventa T(oggetto, etichetta, utente, sorgente).

Ma non è ancora tutto. In alcuni sistemi di etichettatura sociale gli utenti possono "votare" il modo in cui una etichetta descrive o si addice all'oggetto: questo viene fatto mediante un unico voto, negativo o positivo. La relazione T giunge così alla sua forma finale data da T(oggetto, etichetta, utente, sorgente, voto).

7.4.2 Modello a Grafo

In questa visione le folksonomie sono rappresentate come grafi tripartiti. Un grafo, molto semplicemente, è un insieme di elementi o nodi detti vertici, collegati fra loro da archi. Quando si parla di grafi tripartiti, si intende dire che l'insieme dei vertici è suddiviso (partizionato) in tre insiemi (da cui il termine "tripartito"). Tali insiemi sono disgiunti e rappresentano l'insieme degli attori (o utenti), l'insieme dei concetti (i tag) e l'insieme delle istanze (oggetti commentati): un esempio è in figura 7.1. Questi grafi sono interessanti anche perché mostrano le proprietà di quelli che si chiamano "piccoli mondi" (per *small-world network* si intendono tipi di grafi in cui la maggior parte dei nodi non sono vicini ma possono essere raggiunti da qualsiasi altro nodo mediante un piccolo numero di passi): restando in ambito "sociale" tutto ciò ricorda il grande matematico Erdos, che cambiò il modo di lavorare di molti matematici rendendo la matematica una scienza sociale da sviluppare ed elaborare in gruppo. Più formalmente

si parla di “lunghezza caratteristica dei cammini”, che esprime la lunghezza media del cammino minimo fra due nodi e i “coefficienti di raggruppamento”, che esprimono il fatto che due nodi siano o meno direttamente connessi.

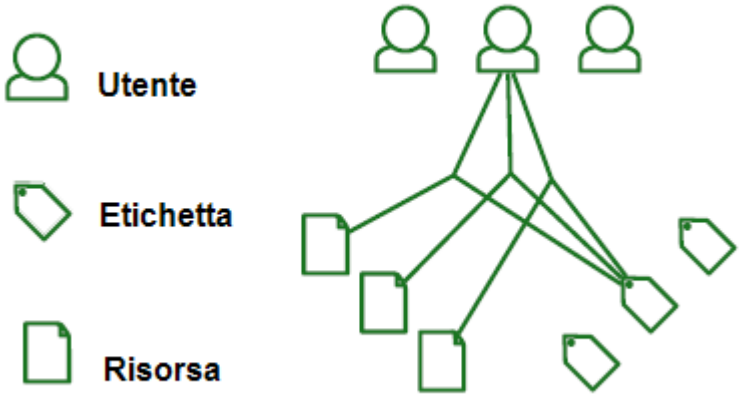

Figura 7.1 – Un grafo tripartito in cui si notano alcune relazioni

Il significato di quanto sopra è che, in media, ogni utente, tag o risorsa, può essere raggiunta con poco sforzo, ossia entro pochi click del mouse (si parla di 3 click) partendo da qualunque punto, anche se le community crescono incorporando milioni di nodi.

Gli utenti attribuiscono dei concetti agli oggetti mediante etichette. Non sono prese in considerazione sorgenti o voti, ma si ricade in un caso simile al precedente, perché è possibile rappresentare le folksonomie come relazioni R(attori, concetti, istanze).

La rappresentazione è fatta mediante **ipergrafi** non orientati dotati di archi ternari, dove ogni arco rappresenta una associazione fra le parti. L’ipergrafo è definito da:

$$G(R) = < V, E >$$

dove

$$V = AUCUI$$

$$E = \{(a, c, i) / (a, c, i) \in T\}$$

con A insieme degli attori, C insieme dei concetti e I insieme delle istanze (l'unione è quella disgiunta)

Per maggiore semplicità questi grafi tripartiti possono essere sostituiti da tre grafi bipartiti (vedere figura 7.2), con archi regolari (non ternari) in cui sono più facilmente rappresentati diversi legami: AC (legami attori-concetti), CI (legami concetti-istanze) e AI (legami attori-istanze).

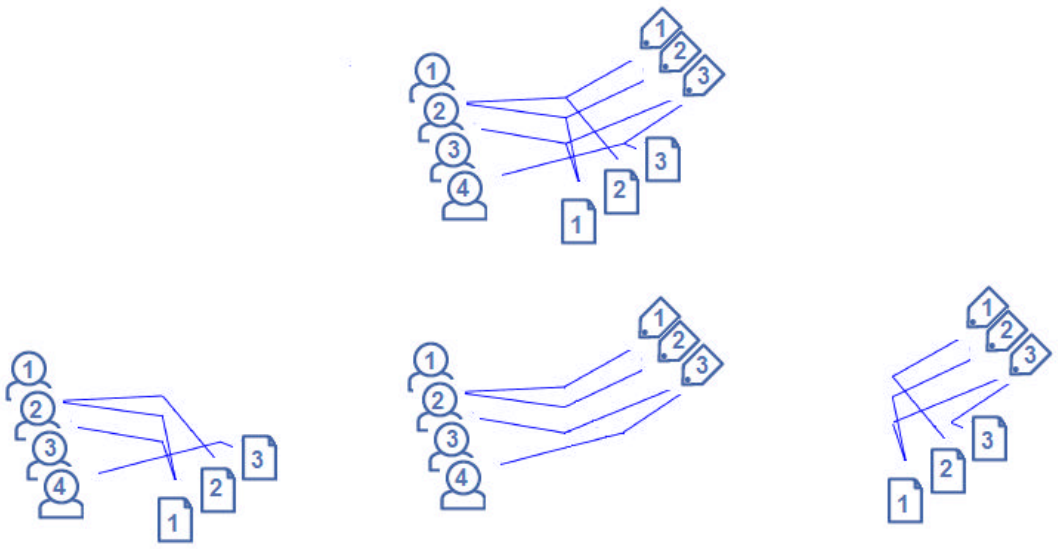

Figura 7.2 – Un grafo tripartito e i tre grafi bipartiti da esso ricavati

Il legame AC riguarda i collegamenti fra persone e concetti da loro usati per etichettare un oggetto; non tutte le sorgenti folksonomiche permettono di collegare un utente all'etichetta da lui scelta, ma, dove permesso, il collegamento può essere dotato di un peso corrispondente al numero di volte che il concetto è stato usato come etichetta (ad esempio, lo stesso utente ha usato tre volte il concetto "età"). Nell'ambito dello studio delle reti sociali, il grafo è conosciuto come "rete di affiliazione" (osservare ancora la fig. 7.2) in cui i pesi indicano la consistenza del legame (ad esempio in figura 7.3 l'arco più marcato mostra un legame più forte). Tale legame è considerato nel caso si vogliano studiare relazioni sociali di utenti che condividono gli stessi interessi. In tutti quei sistemi che non tengono traccia di qualche tipo di legame, non è ovviamente

possibile studiare la rete di affiliazione che ne risulterebbe. Considerazioni analoghe valgono per i legami CI e AI.

Figura 7.3 – Trasformazione dell'ipergrafo e rappresentazione di un legame forte.

Queste concetti li ritroveremo in 8.3 quando si parlerà del portale ArsMeteo, la cui struttura poggia sulle folksonomie e ben si presta ad essere rappresentata dal modello appena descritto.

7.4.3 Modello a Tupla

La descrizione delle folksonomie avviene mediante tuple F = (U, T, R, Y) dove U, T, R sono insiemi finiti di utenti (users), etichette (tags) e risorse (resources); Y è una relazione ternaria fra di essi, cioè Y è un sottoinsieme del prodotto cartesiano U x T x R. Anche in questo caso si ottiene una rappresentazione simile alla prima.

A volte si possono considerare quelle che si chiamano Personomie (restrizioni di F ad un singolo utente).

Da questo modello a tupla è facile passare al modello a grafo, prendendo come insieme dei vertici l'unione disgiunta di U, T ed R e prendendo come insieme degli archi l'unione dei seguenti insiemi:

$$\{(u,t)/\exists r\in R,(u,t,r)\in Y\}\cup\{(t,r)/\exists u\in U,(u,t,r)\in Y\}\cup\{(u,r)/\exists t\in T,(u,t,r)\in Y\}$$

7.5 Dal PageRank al FolkRank

La difficoltà principale è quella di adattare la struttura ad ipergrafo propria delle folksonomie alla struttura a link del web (Hotho at al, 2006).

Una volta ottenuta la struttura desiderata per le folksonomie, dal PageRank è semplice passare al FolkRank, a volte detto SocialRank (Bao et al, 2007). Nella struttura a trigrafo c'è un triarco che unisce tre elementi: un utente, un tag e una risorsa. Si passa ad una struttura a web togliendo il triarco e collegando ogni elemento mediante un arco. La fig. 7.4 rende meglio l'idea.

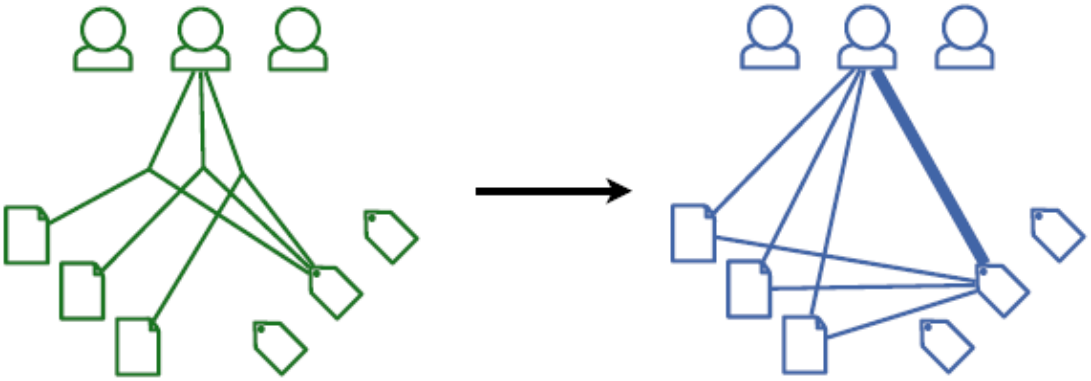

Figura 7.4 – Passaggio da una struttura a trigrafo a una struttura a link

Fatto questo si ottiene una rappresentazione a matrice nella stessa maniera in cui era stato fatto per il web:

$$w^{k+1} = dAw^{k} + (1-d)\,p$$

dove w è il vettore della rilevanza, d esprime come al solito il tempo di navigazione, A è quella che nel PageRank era la matrice G e p è l'approssimazione iniziale. Anche in questo caso valgono considerazioni analoghe a quelle del PageRank.

7.6 Raccolta e recupero delle informazioni

Nei capitoli precedenti dopo aver definito che cosa sono le folksonomie si è espressa una breve considerazione sul tema dell'information retrieval che ci ha portato a fare un'analisi delle differenze tra folksonomie, tassonomie e ontologie. Riprendiamo ora il discorso ricordando che nelle ricerche di informazioni quello che importa ai fini di raccolta e recupero sono le differenze fondamentali di classificazione con cui i dati che costituiscono le informazioni vengono strutturati:

- automatizzazione (tramite gli *spider* dei motori di ricerca);

- classificazioni fatte da utenti umani esperti (ad esempio le *directory*);

- classificazioni fatte da utenti umani non esperti (nel caso delle folksonomie).

In base a queste differenze si può stabilire la qualità degli **IRS** e vedere come l'introduzione delle folksonomie contribuisce nel reperimento di informazioni.

I motori di ricerca sono ormai studiati e testati sin dal loro affacciarsi su Internet. (Morrison, 2007) ha raccolto i risultati relativi ai test sui motori di ricerca e li ha confrontati su test analoghi fatti sui servizi di directory e sulle folksonomie, comprendendo quindi i tre tipi di classificazione: automatica, di esperti e di non esperti. I confronti si sono concentrati sui seguenti siti: Google e MSN (presi come campione rappresentativo dei motori di ricerca e della classificazione automatizzata), i servizi di directory offerti da Altavista e Yahoo (presi come campione per il secondo tipo di classificazione) e Del.icio.us, Furl e Reddit (presi come campione per il terzo tipo di classificazione).

Si è confermato come, per esempio, quando i risultati di ricerche hanno sovrapposizioni nei tre diversi gruppi, tali risultati sono quelli che si rivelano più rilevanti. Considerando invece aspetti più pratici, si è notato come, ad esempio, un sistema singolo possa migliorare la rilevanza dei suoi risultati; il miglioramento si basa su una sorta di "alleanza": in questo senso Folksonomie e Directory sono risultate non compatibili, mentre i motori di ricerca possono essere accoppiati indifferentemente o solo con Folksonomie o solo con Directory. Un'altra interessantissima conseguenza, sempre nell'ambito del miglioramento della rilevanza dei risultati, può essere fatta

considerando il rapporto costi/benefici: chi cercasse di migliorare i propri sistemi di ricerca mediante directory avrebbe un miglioramento di circa il 180% mentre con le folksonomie le prestazioni aumenterebbero di circa il 130%; tuttavia, voler utilizzare le directory ha un costo del 200% rispetto a quello che si avrebbe con le folksonomie perché significa (si pensi per esempio alle tassonomie o alle ontologie) dover creare meccanismi di controllo centralizzato e convocare esperti per trovare e classificare pagine web e pagarli; le folksonomie vedono invece "lavorare" tanti impiegati, già stipendiati, che dedicano, magari anche piacevolmente, piccole quantità di tempo e sforzo per "taggare": questa è già realtà in alcune intranet aziendali, come avviene per IBM, la prima a sperimentare in questo senso.

I risultati delle ricerche di Morrison sono stati fatti anche tenendo conto dei diversi bisogni informativi: ad esempio ricerca di documenti specifici, di documenti riguardanti l'intrattenimento o di notizie. Esisteva l'ipotesi che siccome le folksonomie sono metodi di collezionamento sociale, dove gli insiemi di risorse sono costruite da molti utenti che aggiungono o taggano le risorse di loro interesse, i sistemi folksonomici avessero dei vantaggi sui metodi di ricerca tradizionali per quanto riguarda almeno l'intrattenimento e le news (questo perché un sistema sociale può fare delle aggiunte immediate e non presenta le gap dovute, nel caso dei motori di ricerca, agli intervalli di tempo necessari agli spider per scandire la rete o, nel caso delle directory, ai tempi necessari agli esperti per la catalogazione); tuttavia bisogna fare una precisazione: è vero si che le folksonomie si comportano meglio nel caso di intrattenimento o news, ma non è vero che si comportino meglio dei metodi di ricerca tradizionali. Bisogna altresì precisare che le folksonomie apportano il loro contributo positivo se unite ai motori di ricerca, in quanto insieme producono set di risultati significativamente migliori. Inoltre le folksonomie sono relativamente nuove, mostrano un rafforzamento ogni volta che viene aggiunto un utente e sembrano molto promettenti in diversi ambiti.

7.7 Conclusioni

Le folksonomie sono una delle varie applicazioni nate grazie al Web 2.0, il cui successo, dovuto alla facilità d'uso, ha permesso ai vari siti che le utilizzano di

raggiungere un considerevole numero di utenza in un periodo di tempo relativamente breve.

Si è visto inoltre come si possano trarre vantaggi dall'unione di folksonomie con motori di ricerca e altrettanto con le tassonomie e le ontologie.

Quello che si può prevedere o auspicare per il prossimo futuro è quindi una integrazione sempre più marcata tra diversi sistemi e la produzione di nuove funzionalità che renderanno le folksonomie sempre più ricche.

Capitolo 8
ArsMeteo

I precedenti due capitoli si sono occupati di ciò che può essere definito “Web Sociale” nelle sue varie manifestazioni (Web Semantico e Web 2.0 con le sue “varianti”).

Nel presente capitolo viene invece trattato il portale ArsMeteo che possiede alcune delle caratteristiche descritte nel capitolo 7 mentre il prossimo capitolo illustra l’algoritmo di PageRank adattato alla struttura di ArsMeteo.

8.1 – Cos’è ArsMeteo

ArsMeteo è un “Portale di interesse”, ossia un tipo di portale specializzato, rivolto a tutti quelli che sono attratti dall'argomento trattato (vedere prossimo paragrafo) e quindi a utenti locali (iscritti al sito) e utenti globali (visitatori della rete non registrati). il sito ospita un numero consistente di risorse Internet messe a disposizione del pubblico. A dirla con le parole del web 2.0 si tratta di una applicazione web con lo scopo di aggregatore informazioni e che offre un servizio di navigazione relativamente ai propri contenuti e al proprio interno. Inoltre essendo figlio del Web 2.0 permette di raccogliere, presentare, consultare e condividere le proprie risorse in accordo alle linee del Web 2.0.

8.2 – Un portale di interesse artistico

I contenuti trattati dal portale riguardano una vasta gamma di materiale artistico organizzato in uno spazio virtuale che ha delle analogie con una mostra perennemente disponibile e dinamica, perché il sito viene continuamente arricchito di nuovo materiale.

Le risorse sono fornite da privati o da enti che hanno in comune la curiosità per l'arte e che incrementano nel tempo il numero di opere aggiungendo alla collezione presente altri testi, video, immagini, brani musicali e altre produzioni di genere artistico.

ArsMeteo nasce da Giorgio Vaccarino, architetto sin dal 1969, fra i primi ad utilizzare tecniche multimediali nel contesto artistico. L'idea ha trovato naturale realizzazione tramite l'Associazione Culturale ArsMeteo, partecipante italiana di un più ampio progetto di respiro europeo.

8.3 – Caratteristiche generali di ArsMeteo

Le caratteristiche sono quelle generali dei siti Web 2.0 in cui sia presente la possibilità di associare dei significati ai contenuti.

Le risorse presenti sono dei tre tipi "richiesti": autori, opere e tag e la struttura risultante è quella descritta in 7.4.2. Gli autori producono delle opere che "caricano" all'interno del sito e, nel far questo, associano dei tag descrittivi delle loro creazioni. Successivamente altri autori oppure semplici visitatori di passaggio possono affiancare le loro descrizioni a quelle già esistenti. Tuttavia non è possibile sapere quale autore ha scelto quale tag per descrivere quale opera (questo non permette di fare parte delle elaborazioni di cui si accennava, a proposito della trasformazione da una struttura a grafo tripartito ad una struttura a link, presentate in 7.4.2: per riportare un esempio, dal momento che non viene tenuta traccia di quali tag sono stati usati da quali autori, non è possibile scoprire dei gruppi con interessi comuni tramite analisi dei tag). Si sa solo che le opere sono state prodotte da alcuni autori e, disgiuntamente, che le opere possiedono un certo numero di descrizioni e quindi si possono ottenere soltanto due dei tre grafi bipartiti di cui si diceva sempre in 7.4.2, quelli relativi ai legami utente-risorsa e tag-risorsa (vedere ancora la figura 7.2).

I visitatori possono poi esprimere (oltre che aggiungerne di nuovi, come detto) dei giudizi sull'accuratezza di una tag per descrivere un opera, aumentandone o diminuendone la misura di rilevanza per mezzo dei simboli "+" e "-". Questi fatti forniscono ai legami fra le opere e i tag il carattere della dinamicità, in quanto a seguito dei cambiamenti indotti dalle azioni dei visitatori le associazioni tag-opera (e viceversa)

variano nel tempo. Queste variazioni possono essere utili per osservare le tendenze in atto.

I tag scelti possono essere costruiti mediante una o più parole per esprimere ciò che l'opera in questione richiama alla memoria razionale o emotiva.

8.4 – Homepage: il punto di ingresso del portale

Il portale fornisce buoni stimoli di esplorazione e ricerca al visitatore già dalla home page, dove la prima cosa che attira fortemente l'attenzione è un riquadro contenente una vista satellitare di una porzione della terra in cui sono presenti in ordine sparso e mischiate tra loro delle miniature di opere e delle tag. Questo primo dettaglio serve a stimolare l'esplorazione di opere e tag. L'interfaccia offre inoltre la possibilità di condurre ricerche inserendo parole chiave nella casella di testo fornita.

La grafica è semplice e la lettura e la visualizzazione della home sono abbastanza chiare. L'intuitività e la facilità degli strumenti iniziali di esplorazione e di ricerca è ben progettata e di facile comprensione. La presenza di un numero limitato di colori nel contorno del riquadro rende l'interfaccia gradevole e appropriata, tuttavia la sovrapposizione occasionale di alcune tag e/o opere non permette una loro immediata visione. In conclusione l'organizzazione della struttura interna delle informazioni di progettazione del sito lo rende amichevole agli ospiti (fig 8.1).

Figura 8.1 – La Home Page di ArsMeteo

8.5 – Navigazione

La navigazione fra le risorse può avvenire principalmente per opera o per tag.

Scegliendo un'opera si viene riportati ad una pagina che comprende, dall'alto verso il basso, l'opera scelta, una serie di tag associati (ai quali è possibile incrementare o decrementare l'importanza), una casella di testo in cui inserire, se si vuole, un nuovo tag e infine, eventualmente, le anteprime di opere correlate a quella scelta.

Dato che le opere possono essere descritte solo tramite l'associazione con etichette, non è ovviamente possibile correlare direttamente un'opera con le altre. La correlazione avviene perciò in due semplici passi. Prima si considerano tutti i tag associati all'opera, poi tutte le opere associate ai tag trovati (esclusa l'opera di partenza).

Le opere così ottenute devono essere presentate all'utente in modo che egli abbia la possibilità di vederne più di una contemporaneamente. Questo viene fatto mediante un procedimento che calcola delle miniature e le dispone visualizzandone dodici per

pagina. Se le opere correlate sono meno di dodici, basta una sola pagina. Se sono di più vengono proposte nelle pagine successive, sempre in numero di dodici per pagina.

Nelle parte superiore della pagina in cui viene presentata l'opera scelta trovano un piccolo spazio i dati anagrafici (titolo, autore, mese e anno dell'opera) e più in basso, dopo la casella di testo per inserire tag, trovano posto i dati storici (data di inserimento dell'opera, numero di visite e interventi) (fig 8.2). Per indicare che all'opera considerata sono associate informazioni supplementari, sotto all'opera stessa viene posta la parola "approfondimenti", cliccando la quale viene presentata la documentazione collegata.

Scegliendo un tag, si è riportati ad una pagina che mostra, da sinistra verso destra, anteprime delle opere collegate e altri tag (questi ultimi suddivisi in due categorie: in alto e rinchiusi in un riquadro, quelli che sono associati in modo lessicale al tag di partenza, ossia quelli che, se presenti, contengono al loro interno il tag richiesto, come prefisso o postfisso; sempre dentro al riquadro trovano posto, se presenti, anche i sinonimi, mentre in basso sono riportati quei tag che sono correlati a quello di partenza). Nella parte superiore sono riportati alcuni link a forma di icone, cliccando le quali si può avere cercare quel tag in altri siti (fig.8.3).

Da notare che in entrambe le modalità di navigazione è possibile scegliere se continuare a navigare per tag oppure cambiare modalità e navigare per opera (e viceversa).

Funzione interessante è quella data dal collegamento "Tag" nell'header delle pagine, cliccando il quale viene visualizzata una pagina contenente quella che si chiama "nuvola di tag", cioè una rappresentazione dei tag, in ordine casuale, con differente grandezza dei caratteri per sottolineare il loro grado di importanza (più grandi sono le lettere, più "usato" è il tag): per avere un'idea del concetto di "importanza di un tag" vedere la parte destra della figura 8.3.

E' poi possibile, selezionando da "menu", condurre delle ricerche per generi (ad esempio fotografia, poesia, scultura, incisione,…), autori (visualizzabili per numero di opere, per ordine alfabetico o per ordine cronologico) ed opere (in questo caso sono possibili diverse combinazioni dei parametri di ricerca).

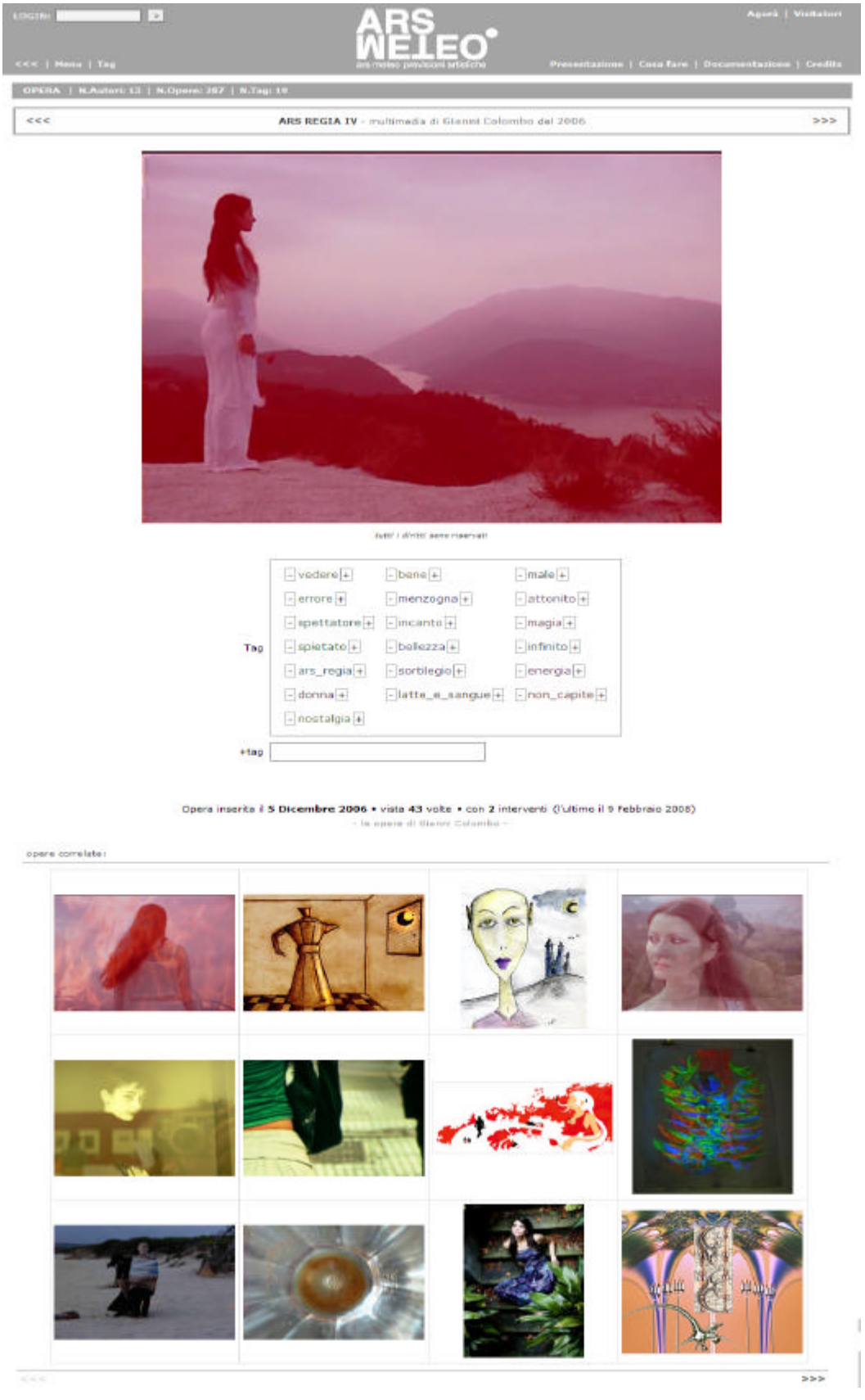
ARS WEIEO
Presentazione | Cosa fare | Documentazione | Credits
<<< | Mappa | Tag
OPERA | N.Autori: 13 | N.Opere: 387 | N.Tag: 19
<<<
ARS REGIA IV - multimedia di Gianni Colombo del 2006
>>>
Tutti i diritti sono riservati
Tag
vedere
bene
male
errore
menzogna
attonito
spettatore
incanto
magia
spiatato
bellezza
infinito
ars_regia
sortilegio
energia
donna
latte_e_sangue
non_capite
nostalgia
+tag
Opera inserita il 5 Dicembre 2006 • vista 43 volte • con 2 interventi (l'ultimo il 9 febbraio 2008)
- le opere di Gianni Colombo -
opere correlate:
<<<
>>>

Figura 8.2 – Navigazione per opera

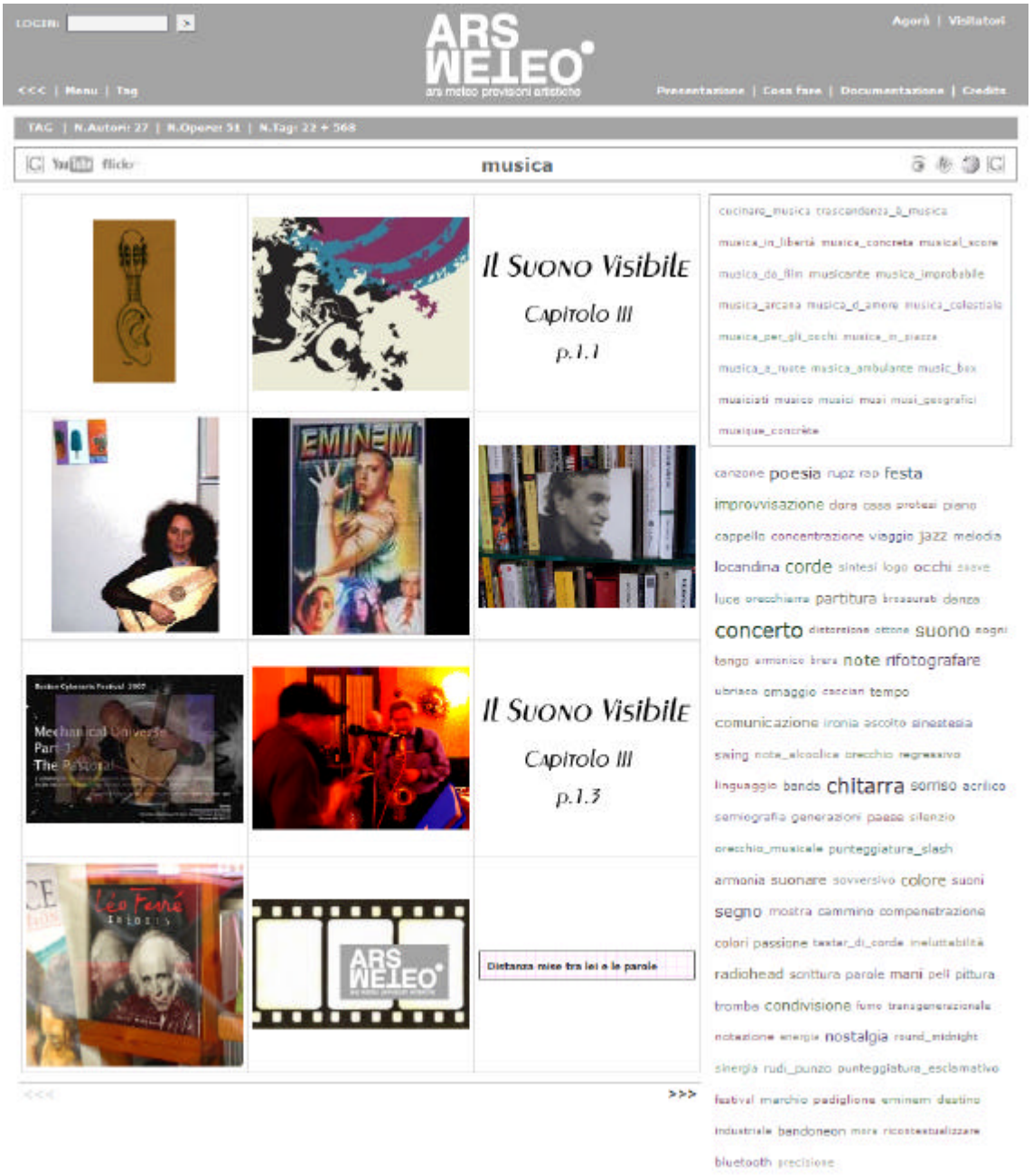

Figura 8.3 – Navigazione per tag

8.6 – Registrazione, esempi di utenti, possibili usi

Una particolarità è la registrazione degli utenti che vogliano inserire opere (o perché autori o perché rappresentanti degli autori, dai quali siano stati autorizzati). Infatti per fare operazioni di inserimento nel database è indispensabile un identificativo che deve essere fornito da Arsmeteo: la procedura di "registrazione" non avviene, come per la maggior parte di siti, immettendo un username e una password nei campi destinati ad accogliere queste informazioni, ma si fa contattando direttamente gli amministratori del sistema mediante una mail a cui inviare i propri dati. Questa procedura è un po' inusuale, ma da' senz'altro un tocco di serietà professionale (sia artistica che informatica) a quanto il portale si propone di offrire.

Esempi di utenti sono: artisti che vogliano avere una idea dell'andamento delle tendenze in atto, curatori che intendano programmare delle mostre (per rintracciare opere e autori che meglio possano rappresentarle) o galleristi.

Un esempio di uso educativo: possono trarne vantaggio studenti e docenti di discipline artistiche, facendo uso di documentazioni e approfondimenti presenti nel sito.

8.7 – Risorse a supporto

Le risorse hardware e software sono curate e ospitate presso easybit (http://www.easybit.it/). L'hardware è costituito da un Server HP con processore AMD da 1.8 Ghz, 2GB di RAM e un totale di memoria di massa di 660 GB suddiviso in tre dischi rigidi. Le risorse software sono date dalla combinazione conosciuta con l'acronimo LAMP per il lato server (per il lato client sono indicati JavaScript, Flash, una fra le seguenti versioni browser IE6, Mozilla 1, Opera 8, Safari 2 o superiori e Adobe Flashplayer 8 o superiore). L'interrogazione di ArsMeteo produce una serie di documenti web dinamici in un formato standard del tipo HTML/XHTML che saranno visualizzati dal browser.

Le risorse umane coinvolte sono costituite da docenti, ricercatori e studenti del Dipartimento di Informatica dell'Università di Torino (http://www.di.unito.it) e da programmatori della web agency Easybit e da artisti torinesi.

8.8 – Alcuni dati e osservazioni sulla crescita di ArsMeteo

In questo paragrafo parleremo di crescita intendendo l'incremento del numero di risorse ospitate nei mesi passati.

Nella home di ArsMeteo è indicato il numero di risorse presenti in un dato momento. Si sono confrontati i dati contenuti in immagini della Home presenti in archivi fra loro e fra i dati contenuti visualizzando la Home on line.

La seguente Tabella mostra i dati relativi (in ultima colonna i totali):

Mese/Anno	Autori	Opere	Tag	Risorse
09/2007	72	1192	5132	6396
12/2007	104	1675	8061	9840
03/2008	127	2414	15696	18237
06/2008	152	3045	20701	23898

Tabella 8.1 – Distribuzione del numero di risorse in tre successivi intervalli di tempo

La seguente tabella mostra invece i dati relativi alla crescita percentuale nei trimestri indicati

	09/2007 12/2007	12/2007 03/2008	03/2008 06/2008
Autori	44.44	22.11	19.68
Opere	40.52	44.11	26.13
Tag	57.07	94.71	31.88

Risorse	53.84	85.33	31.04

Tabella 8.2 – Crescita percentuale per trimestre

La prima riga mostra un certo decremento nel tasso di crescita della partecipazione di artisti al sito (occorrerebbero attività e programmi di pubblicità del sito e reclutamento di artisti), mentre la produttività delle opere ha continuato a crescere, a dispetto del decremento nel numero di iscrizioni di artisti. Nel secondo trimestre analizzato si nota un picco nell'attività di etichettatura (dato che lo spazio di opere è minore allo spazio delle tag, nel trimestre in esame vengono associate più tag a singole opere).

8.9 – Conclusioni

L'arte on-line, organizzata in mostre, musei e gallerie virtuali è una gran cosa per tutte quelle persone che, interessate all'argomento, per motivi di lavoro e di tempo o per altri motivi, non possono visitare le controparti reali. E' inoltre molto utile per tutte quelle attività educative e di studio che coinvolgano gruppi "del mestiere".

Tuttavia non bisogna dimenticare che una mostra dal vivo, ad esempio, è tutta un'altra cosa, coinvolgendo anche il contatto sociale con altre persone che partecipano; la realtà virtuale non può ben rendere: così come è un potentissimo strumento per costruire, lo è altrettanto per distruggere la bellezza di un'opera goduta dal vivo.

Capitolo 9
L'applicazione: descrizione e listato

In questo capitolo si presenta il programma per il calcolo del PageRank adattato alla struttura di ArsMeteo. Si descrivono gli strumenti utilizzati e le soluzioni adottate.

9.1 Il software utilizzato

Durante l'attività di scrittura del sorgente, il linguaggio di programmazione utilizzato è stato Java nella sua versione 1.5.0.04. Non è stato necessario servirsi ambienti di sviluppo complessi e l'editor scelto, il semplice TextPad 4.5.0, è stato più che sufficiente nel supporto alla programmazione.

Il sistema operativo ospite è stato in un primo momento windows xp home edition, sul quale sono stati installati sia TextPad che l'SDK Java 1.5, senza incontrare nessuna difficoltà.

Altri elementi necessari sono stati il Server db MySql versione 5.0.26 per win32 e il mysql-connector-java-5.1.6. L'installazione del Server MySql non ha dato nessun problema, mentre il connector in un primo momento si è rivelato difficile da trattare: i problemi sono stati risolti mettendolo nella directory jdk1.5.0_04\jre\lib\ext (il tutto senza scomodare variabili d'ambiente come CLASSPATH).

Non appena il programma ha cominciato a dare i primi risultati, si è pensato di avvicinarsi di più al tipo di macchina (quella di produzione descritta nel capitolo 8) su cui avrebbe dovuto eseguire la classe. Si è così installata una versione Server di VMWare e si è virtualizzata una macchina Debian nella versione Etch 4.0 su cui sono stati installati Java 1.5.0, il Server MySql e il connector java: in questo caso il connector è stato copiato nel percorso usr/lib/java ed è stato modificato il file .bashrc impostando la variabile CLASSPATH e indicandone poi l'esportazione.

In entrambe le due configurazioni si è poi dovuto indicare a MySql di presentare i risultati di alcune operazioni con una certa precisione, richiedendo quindi un output comprendente almeno 12 cifre decimali. Questo è stato fatto aggiungendo una linea (set div_precision_increment = 12;) nel file di configurazione (chiamato my.ini in ambiente windows e my.cnf in ambiente linux).

Altro software utilizzato è stato octave 3.0, per controllare alcuni risultati.

9.2 Le tabelle di ArsMeteo.

Fra tutte le tabelle del database ne sono state utilizzate in tutto quattro: quelle relative a opere, tag, relazioni fra tag e opere e quella prodotta allo scopo di poter facilmente costruire la matrice necessaria per il calcolo del FolkRank.

Le tabelle sono riportate in fig. 9.1, dove si vedono anche i legami fra di esse (in successivi paragrafi verranno descritte le query che necessitano di una spiegazione).

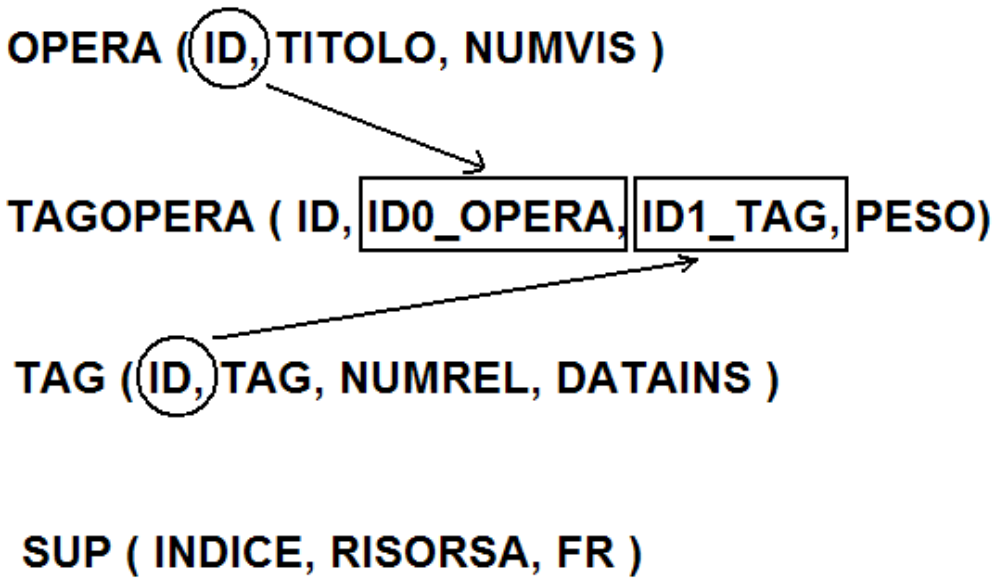

Figura 9.1 – Le tabelle utilizzate

9.3 Classi prodotte

Le classi essenziali sono quattro e sono necessarie per descrivere l'elemento di una matrice e la matrice stessa, per gestire la connessione al database e ricavare le informazioni necessarie al calcolo e il main.

La descrizione di classi e metodi è contenuta all'interno di ogni classe, nello stile dei commenti java. Tali commenti sono quelli utilizzati dall'utility javadoc per produrre la documentazione di supporto ai programmi.

Le immagini prodotte in questo capitolo servono per illustrare alcuni concetti e non sono incluse nella documentazione prodotta da javadoc.

9.3.1 La classe ElementoMatriceSparsa

```
/**
    Le matrici sparse sono quelle in qui la maggior parte di elementi ha lo
    stesso valore, detto valore dominante.
    E' quindi possibile rappresentare queste matrici considerando solo gli
    elementi diversi da quello dominante.
    Come conseguenza, nelle applicazioni che lo richiedono, è possibile un
    considerevole risparmio di memoria.
    Gli elementi di una matrice sparsa sono quindi rappresentati ognuno
    da una terna: due interi (l'indice di riga e l'indice di colonna) e un
    double (il valore).
    Una matrice sparsa sarà rappresentata invece da un array contenente
    oggetti di questa classe.
    Per indicare un oggetto di questa classe nella documentazione verrà
    utilizzata l'abbreviazione EMS.
*/
public class ElementoMatriceSparsa{

    //variabili di classe con nomi autodescrittivi
    private int indiceRiga;
```

```
private int indiceColonna;
private double valore;

//costruttore
/**
		Costruisce un elemento di una matrice sparsa.
	@param unaRiga l'indice di una riga.
	@param unaColonna l'indice di una colonna.
	@param unValore il valore dell'elemento corrispondente a riga e
		colonna forniti.
*/
public ElementoMatriceSparsa(int unaRiga, int unaColonna, double unValore){
	indiceRiga = unaRiga;
	indiceColonna = unaColonna;
	valore = unValore;
}

//metodi con nomi autodescrittivi
/**
		Dato un oggetto EMS, ne restituisce l'indice di riga.
	@return indiceRiga, l'indice della riga in cui si trova l'elemento.
*/
public int restituisciIndiceRiga() { return indiceRiga; }

/**
		Dato un oggetto EMS, ne restituisce l'indice di colonna.
	@return indiceColonna, l'indice della colonna in cui si trova l'elemento.
*/
public int restituisciIndiceColonna() { return indiceColonna; }

/**
		Dato un oggetto EMS, ne restituisce il valore.
```

```
        @return valore, il valore dell'elemento.
*/
public double restituisciValore() { return valore; }

/**
                Dato un oggetto EMS, ne imposta l'indice di riga.
        @param nuovoIndiceRiga il nuovo indice per la riga.
*/
public void impostaIndiceRiga(int nuovoIndiceRiga) {
        indiceRiga = nuovoIndiceRiga;
}

/**
                Dato un oggetto EMS, ne imposta l'indice di colonna.
        @param nuovoIndiceColonna il nuovo indice per la colonna.
*/
public void impostaIndiceColonna(int nuovoIndiceColonna) {
        indiceColonna = nuovoIndiceColonna;
        }

/**
                Dato un oggetto EMS, ne imposta il valore.
        @param unValore il nuovo valore.
*/
public void impostaValore(double unValore) { valore = unValore;}

/**
                Sovrascrive il metodo toString per adattarne l'output.
*/
public String toString(){
        return "(" + indiceRiga + " " + indiceColonna + " " + valore + ")";
}
```

}

9.3.2 La classe MatriceSparsa

```
import java.io.*; // Per stampa su file di vettore e matrice

/**
    Questa classe rappresenta le matrici sparse.
    Le informazioni necessarie per tale rappresentazione sono un array di elementi della classe ElementoMatriceSparsa, un intero indicante il numero di righe della matrice, un intero con analogo significato per le colonne e un altro intero che indica quanto è grande la rappresentazione della matrice (è la dimensione dell'array contenente gli elementi non dominanti).
    La rappresentazione dei vettori è la stessa di quella delle matrici.
    I metodi della classe sono quelli necessari per calcolare dei valori usando con il metodo delle potenze:
  - prodotto di uno scalare per un vettore
  - somma di due vettori colonna
  - somma di due vettori riga
  - prodotto di un vetore colonna per un vettore riga
  - prodotto di un vettore riga per una matrice sparsa
    Gli altri metodi definiti, sono a supporto dei precedenti o a supporto del metodo delle potenze.
    Nella documentazione il termine MatriceSparsa viene abbrevia in MS
*/

public class MatriceSparsa {

  private ElementoMatriceSparsa[] matriceSparsa;
  private int numeroRighe;
  private int numeroColonne;
  private int dimRappresentazione;
```

```
/**
    Costruisce una matrice vuota
*/
public MatriceSparsa(){ matriceSparsa = null; }

/**
    Costruisce una matrice a partire dagli elementi (forniti) che la    compongono.
  @param unVettoreDiElementi un vettore di elementi di tipo EMS
  @param numRighe il numero di righe della matrice originaria (di tipo int).
  @param numColonne il numero di colonne della matrice originaria (di tipo int).
  @param dimensione la grandezza della rappresentazione (di tipo int).
*/
public MatriceSparsa(
  ElementoMatriceSparsa[] unVettoreDiElementi,
  int numRighe,
  int numColonne,
  int dimensione){
    matriceSparsa = unVettoreDiElementi;
    numeroRighe = numRighe;
    numeroColonne = numColonne;
    dimRappresentazione = dimensione;
}

/**
     Restituisce un intero rappresentante il numero di colonne della MS
    @return numeroColonne, il numero di colonne della MS
*/
public int getNumColonne() { return numeroColonne; }

/**
     Restituisce la MS
```

```
     @return matriceSparsa, la MS
  */
  public ElementoMatriceSparsa[] getMatriceSparsa() { return matriceSparsa; }

  /**
       Visualizza una matrice sparsa sul display.
     Mediante il formatter viene deciso quante cifre decimali visualizzare.
      Si è scelto di visualizzarne otto. Il metodo è indicato per i vettori,       in quanto non
visualizza gli indici di riga e colonna.
  */
  public void visualizzaSparsa() {
    System.out.println(dimRappresentazione);
    java.text.NumberFormat formatter =
     java.text.NumberFormat.getNumberInstance();
    formatter.setMaximumFractionDigits(8);
    formatter.setMinimumFractionDigits(8);
    for(int i = 0; i < dimRappresentazione; i++){
     System.out.print(i + " - ");
     System.out.print(
      matriceSparsa[i].restituisciIndiceRiga() +
      " " +
      matriceSparsa[i].restituisciIndiceColonna() +
      " ");
     System.out.println(
      formatter.format(matriceSparsa[i].restituisciValore()) + " ");
    }
    System.out.println();
  }

  /**
     Uno dei metodi principali.
     Dati uno scalare ed un vettore, restituisce il vettore risultato.
```

```
    Definisce una variabile double prodotto in cui mettere il prodotto   dello scalare dato
con il valore di un elemento della matrice sparsa.
   @param scalare uno scalare (di tipo double)
   @param vettore un vettore di tipo MatriceSparsa.
   @return vett, il vettore risultato di tipo MatriceSparsa.
 */
 public static MatriceSparsa prodottoScalare(
  double scalare, MatriceSparsa vettore) {
    // conterrà il valore del prodotto (scalare per valore corrente)
    double prodotto;
    // elemento corrente
    ElementoMatriceSparsa ems;
    // array degli EMS per la MS: bisogna darne la dimensione
    ElementoMatriceSparsa[] ems_prodotto =
     new ElementoMatriceSparsa[vettore.dimRappresentazione];
    // la MS che accoglierà il vettore risultato
    MatriceSparsa vett;
    // scorre il vettore dato
    for(int i = 0; i < vettore.dimRappresentazione; i++){
     // valore dell'elemento del vettore risultato
     prodotto =
       vettore.matriceSparsa[i].restituisciValore() * scalare;
     // EMS corrente, con stessi indici di riga e colonna del
     // vettore dato, ma con valore dell'elemento pari al
     // prodotto dello scalare dato per l'elemento corrente
     // del vettore dato
     ems = new ElementoMatriceSparsa(
       vettore.matriceSparsa[i].restituisciIndiceRiga(),
       vettore.matriceSparsa[i].restituisciIndiceColonna(),
       prodotto);
     // EMS al passo precedente, in posizione i del vettore risultato
     ems_prodotto[i] = ems;
```

```
    }
    vett = new MatriceSparsa( // costruzione della MS risultato
      ems_prodotto,
      vettore.numeroRighe,
      vettore.numeroColonne,
      vettore.dimRappresentazione);
    return vett;
}

/**
    Dati due vettori riga, ne restiuisce la somma.
    Semplicemente si sommano i termini corrispondenti con unciclo for.
  @param v1 un vettore riga di dimensione (1,n) (di tipo MatriceSparsa).
  @param v2 un vettore riga di dimensione (1,n) (di tipo MatriceSparsa).
  @return newMatriceSparsa, un vettore riga somma dei vettori dati.
*/
public static MatriceSparsa somma2vettoriR(
  MatriceSparsa v1, MatriceSparsa v2){
    // Le variabili per la costruzione del risultato
    ElementoMatriceSparsa ems;
    ElementoMatriceSparsa[] ems_ris =
      new ElementoMatriceSparsa[v1.numeroColonne];
    // Scandisce i due vettori, sommandone i valori degli elementi corrispondenti
    for(int i = 0; i < v1.numeroColonne; i++) {
      ems = new ElementoMatriceSparsa(
        0, // sempre zero, perché vettore riga
        i, // colonna corrente
        v1.matriceSparsa[i].restituisciValore() +
        v2.matriceSparsa[i].restituisciValore());
      ems_ris[i] = ems;
    }
  // Costruzione del risultato
```

```
    return new MatriceSparsa(
      ems_ris, 1, v1.numeroColonne, v1.numeroColonne);
  }

  /**
       Prodotto di un vettore riga per una matrice in base alla formula
      M(ij) = W[k+1](j) =+ W[k](i)M(ij)
      @param vR un vettore di tipo MS
      @param mS una MS
      @return newMatriceSparsa, il risultato del prodotto (di tipo MS)
  */
  public static MatriceSparsa prodottoVeloce_vRmS(
          MatriceSparsa vR, MatriceSparsa mS) {
    int cursore = 0;
   // opere e tag in ordine come da tabella SUP
    double[] wk1 = new double[vR.numeroColonne];
    while(cursore < mS.dimRappresentazione) {
      wk1[mS.matriceSparsa[cursore].restituisciIndiceColonna()] +=
        mS.matriceSparsa[cursore].restituisciValore() *
vR.matriceSparsa[mS.matriceSparsa[cursore].restituisciIndiceRiga()].restituisciValore()
;
      cursore++;
    }
    cursore = 0;
    ElementoMatriceSparsa ems;
    ElementoMatriceSparsa[] ems_wk1 =
       new ElementoMatriceSparsa[vR.numeroColonne];
    while(cursore < wk1.length){
      ems = new ElementoMatriceSparsa(
        0, // wk1 ha una sola riga
        cursore,
        wk1[cursore]);
```

```
    ems_wk1[cursore] = ems;
    cursore++;
   }
  // dimRap = #Colonne
   return new MatriceSparsa(ems_wk1, 0, vR.numeroColonne, vR.numeroColonne);
 }


 /**
      Controlla se due vettori soddisfano alla condizione di convergenza.
      Se la differenza di tutti gli elementi di indice corrispondente è minore della quantità eps, allora il test è soddisfatto. Altrimenti  bisogna continuare con il metodo delle potenze.
     @param vR1 il primo vettore
     @param vR2 il secondo vettore
     @return convergenza, un valore di verità
 */
 public static boolean converge(MatriceSparsa vR1, MatriceSparsa vR2, double eps) {
   boolean convergenza = true;
   for(int i = 0; i <  vR1.dimRappresentazione; i++){
     double comp_v1 = vR1.matriceSparsa[i].restituisciValore();
     double comp_v2 = vR2.matriceSparsa[i].restituisciValore();
     double diff = comp_v1 - comp_v2; //  DIVIDENDO
     double abs_comp1 = java.lang.Math.abs(comp_v1);
     double abs_comp2 = java.lang.Math.abs(comp_v2);
     double max_abs = java.lang.Math.max(abs_comp1, abs_comp2); // DIVISORE
     double div = diff / max_abs;
     if (div <= eps) {
       convergenza = true;
       // continua a controllare la convergenza degli altri elementi
     }
     else {
```

```
      convergenza = false;
      break;
    }
  }
  return convergenza;
 }

 /**
      Dato un vettore, controlla che sia un vettore di probabilità, ossia  che la somma delle sue componenti sia uguale all'unità. Se non lo è, stampa la somma delle componenti e restituisce false.
   @param vettoreProbabilita un vettore riga (di tipo MatriceSparsa).
   @return somma1, true o false (di tipo boolean)
 */
 public static boolean normalizzato(MatriceSparsa vettoreProbabilita){
  boolean somma1 = false;
  double somma = 0.0;
  for(int i = 0; i < vettoreProbabilita.dimRappresentazione; i++)
    somma += vettoreProbabilita.matriceSparsa[i].restituisciValore();
  if (java.lang.Math.abs(1 - somma) <= java.lang.Math.pow(10, -12)) {
    somma1 = true;
    System.out.print(somma);
    return somma1;
  }
  else {
    somma1 = false;
    System.out.print(somma);
    return somma1;
  }
 }

 /**
```

```
     Come il precedente, ma senza visualizzazione della somma delle componenti
 */
 public static boolean normalizzato1(MatriceSparsa vettoreProbabilita){
  boolean somma1 = false;
  double somma = 0.0;
  for(int i = 0; i < vettoreProbabilita.dimRappresentazione; i++)
   somma += vettoreProbabilita.matriceSparsa[i].restituisciValore();
  if (java.lang.Math.abs(1 - somma) <= java.lang.Math.pow(10, -12)) {
   somma1 = true;
   //System.out.println(somma);
   return somma1;
  }
  else {
   somma1 = false;
   //System.out.println(somma);
   return somma1;
  }
 }

 /**
     Una matrice è stocastica se la somma di ogni riga è l'unità.
    Controlla se H è sottostocastica percorrendola una sola volta.
     Se la matrice non è ordinata (cioè non è nella maniera in cui è stata costruita), per avere una risposta negativa sulla sotto stocasticità bisogna per forza arrivare fino al fondo, perchè non essendo ordinata non si sa se ci sono altri elementi della stessa riga da sommare.
    @param unaMatrice un oggetto di tipo MS
    @return sottostocastica, un valore di verità
 */
 public static boolean sottostocastica(MatriceSparsa unaMatrice) {
  boolean sottostocastica = false;
  double sommaRiga = unaMatrice.matriceSparsa[0].restituisciValore();
```

```
 int k = 0;
 for(k = 1; k < unaMatrice.dimRappresentazione; k++) {
  // se (due) elementi successivi hanno lo stesso indice
  // allora appartengono alla stessa riga e si possono
  // sommare i valori
  if (unaMatrice.matriceSparsa[k].restituisciIndiceRiga() ==
    unaMatrice.matriceSparsa[k-1].restituisciIndiceRiga()) {
      sommaRiga += unaMatrice.matriceSparsa[k].restituisciValore();
    }
  else {
    if (java.lang.Math.abs(sommaRiga - 1) < 10E-12) {
      sottostocastica = true;
      sommaRiga = unaMatrice.matriceSparsa[k].restituisciValore();
    }
    else {
      sottostocastica = false;
      break;
    }
  }
 }
 // all'uscita dal ciclo c'è l'ultima somma da controllare
 // sia se H[k] == H[k-1] sia se H[k] != H[k-1]
 if (java.lang.Math.abs(sommaRiga - 1) < 10E-12)
  sottostocastica = true;
 else sottostocastica = false;

 return sottostocastica;
}

/**
     Moltiplica un vettore riga per un altro vettore riga.
   Il primo vettore riga è di tipo MatriceSparsa;  il secondo vettore è di tipo int[].
```

```
    Questa moltiplicazione "mista" è stata fatta per limitare le richieste di memoria.
    @param w un vettore riga (di tipo MatriceSparsa).
    @param a un array di interi (di tipo int[])
    @return mRis, un vettore riga (di tipo MatriceSparsa)
  */
  public static MatriceSparsa prodotto_vRa(MatriceSparsa w, int[] a) {
    // Ogni elemento del risultato sarà costituito dalla somma
    // degli elementi di w puntati dagli elementi di a: si prendono
    // e se ne fa la somma
    double wi = 0.0;
    for(int i = 0; i < a.length; i++)
      wi += w.matriceSparsa[a[i]].restituisciValore();
    // La somma così ottenuta si divide per n, cioè per il numero
    // (tag+opere) = w.numeroColonne
    double win = 0.0;
    // restituisce il vettore costituito da numeroColonne elementi
    // tutti uguali; questo vettore dovrà poi essere moltiplicato
    // per alfa e sommato agli altri due, come in FR
    ElementoMatriceSparsa ems;
    ElementoMatriceSparsa[] emsRis =
      new ElementoMatriceSparsa[w.numeroColonne];
    for(int j = 0; j < w.numeroColonne; j++) {
      win = wi / w.numeroColonne;
      ems = new ElementoMatriceSparsa(0, j, win);
      emsRis[j] = ems;
    }
    MatriceSparsa mRis =
      new MatriceSparsa(emsRis, 1, w.numeroColonne, w.numeroColonne);
    return mRis;
  }
}
```

9.3.3 La classe ArsConn

La classe ArsConn si occupa della connessione al database e del reperimento delle informazioni per poter costruire tabelle e vettori.

Dato che il programma non deve accedere ad un database esterno, ma esegue sulla stessa macchina in cui esegue MySQL, si sono usati, in java, gli Statement invece dei PreparedStatement.

Prima della presentazione di questa classe alcuni esempi e spiegazioni serviranno per chiarire le parti più complesse.

Per una migliore comprensione della classe occorre anche rivedere quanto scritto nel secondo capitolo.

9.3.3.1 Database, inconsistenze e metodo checkDB()

Facendo riferimento alla fig. 9.1, le inconsistenze che non devono presentarsi nella base di dati sono due: non ci possono essere valori di tagopera.id1_tag che non sono in tag.id e analogamente non ci possono essere valori di tagopera.id0_opera che non sono in opera.id. Il metodo controlla che non vi siano di questi casi: se ne incontra inibisce la costruzione dell'oggetto ArsConn e termina il programma (se non ci fosse questo controllo, si dovrebbe aspettare per tutto il tempo necessario a costruire la matrice per poi vedere il programma terminare con una eccezione).

Altri casi che possono presentarsi (ma che non sono inconsistenze) sono quelli relativi alle opere o ai tag pendenti. I motivi per cui possono presentarsi queste situazioni possono essere principalmente due: per le opere può essere l'autore stesso a richiederne la rimozione, mentre per i tag ciò può essere deciso perché sono tag che contengono errori oppure tag osceni.

Ad esempio, togliendo un tag perché non accettabile possono restare delle opere a cui non è associato nessun tag. Così si genera il problema delle opere pendenti. Il metodo controlla anche questo caso, ma permette la continuazione dell'esecuzione.

Dato che vi sono quattro possibili casi che possono presentarsi (opere pendenti, tag pendenti, inconsistenza nella tabella dei tag e inconsistenza nella tabella delle opere) si sono esaminate una per una le varie combinazioni che possono occorrere (sedici in tutto) e si è testato il comportamento del programma riproducendole tutte, a partire da una base di dati minima su cui sono state riprodotte le varie eventualità.

Una situazione stranissima che si è verificata durante la scrittura del programma è stata la presenza, fra i tag, di una tupla con valore del campo numrel uguale a uno ma non presente nella tabella tagopera (cioè, la tabella tag avverte che questo tag ha esattamente una relazione con un'opera presente nel database ma poi effettivamente questa relazione non esiste, perché non presente nella tabella delle relazioni). La seguente query

```
select id from tag where numrel = 1 and id not in (select id1_tag from tagopera);
```

restituisce infatti un set di risultati non vuoto e con cardinalità uguale a uno.

Nel prossimo paragrafo si tratta delle pagine pendenti: la fig 9.2 descrive visivamente una situazione di questo genere.

9.3.3.2 Esempio visivo delle pagine pendenti e metodi getDimRapp() e getH()

Per avere un insieme piccolo di opere e tag su cui lavorare, supponiamo che esistano le seguenti relazioni:

In questo modo si ottiene il seguente insieme di tuple, che è adatto allo scopo:

id0_opera	id1_tag	peso
940	4184	1.000
940	4186	1.000
1200	5033	1.000
1200	5034	1.000
185	709	1.000
185	872	1.000

La rappresentazione visiva è quella in fig. 9.2 (a), mentre nella fig. 9.2 (b) si presenta una situazione in cui vi sono opere pendenti. Più avanti, quando si presenterà la classe, si avrà modo di vedere i metodi getDimRapp() e getH(). Il primo calcola la dimensione della rappresentazione della matrice H e per far questo conta le relazioni che le opere hanno con i tag (sei in questo esempio) e poi le relazioni che i tag hanno con le opere (altri sei): per questo motivo all'interno del metodo è presenta una moltiplicazione che raddoppia il numero delle relazioni. Il secondo costruisce effettivamente la matrice H, ricavandola dalle relazioni fra opere e tag. Per far questo, la costruisce in due tempi: la prima parte considerando le opere (parte superiore destra della matrice) e la seconda parte considerando i tag (parte inferiore destra della matrice).

La costruzione della matrice avviene ordinando in ordine crescente di id prima tutte le opere e poi tutti i tag, associando agli id così ordinati quelli che saranno gli indici della matrice, calcolando i valori da mettere nella matrice. Una spiegazione visiva è contenuta nella fig 9.3 (una trattazione più precisa è fatta in 9.3.3.3, dove si descrivono le query più complesse che sono utilizzate nella classe).

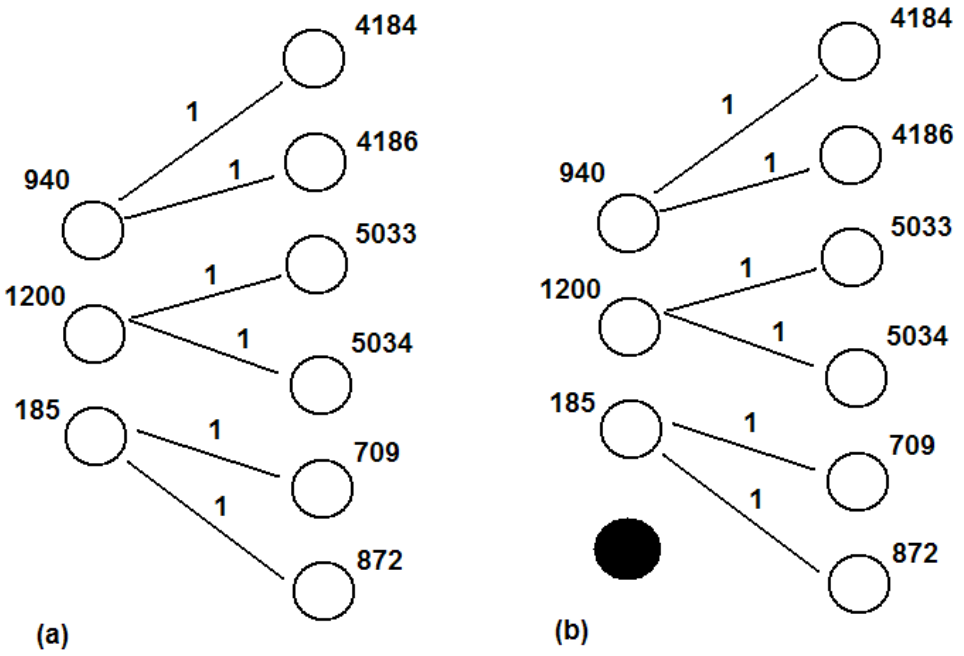

Figura 9.2 Un insieme di relazioni senza pagine pendenti (a) e un insieme di relazioni con un'opera pendente (b).

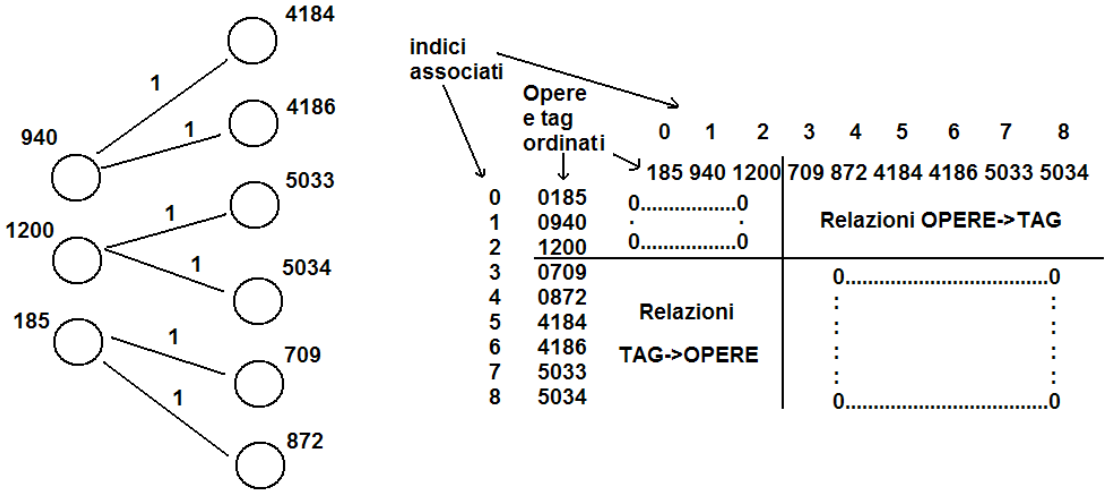

Figura 9.3 Trasformazione del grafo delle relazioni in matrice

9.3.3.3 Le query più corpose e il metodo getH()

Lo scopo delle query è quello di costruire la seguente tabella senza scriverla nel database. Tale tabella viene costruita ogni volta che si avvia il programma di calcolo del FR (per tale query sono necessari diversi minuti). In questa tabella le tuple (i, j, v) significano che nella matrice H il valore v sarà nella posizione indicata da (i, j). Le cifre decimali considerate sono sufficienti per il calcolo richiesto.

Per ogni tabella di esempio si mostrano solo poche righe.

Esempio di tabella che si vuole ottenere:

i	j	v
0	2765	0.0808342
0	2766	0.0808342
0	2767	0.1027402
0	2768	0.0808342
0	2769	0.0808342
0	2770	0.0808342
0	2771	0.0808342
0	2772	0.0808342
0	2773	0.0808342
0	2774	0.0808342

Le tabelle disponibili per calcolare la precedente sono soltanto le seguenti:

opera

id	titolo	numvis
10	IN THE NAME OF THE CHILD	33
27	GEORGE W. BUSH & COLIN POWELL	55
28	GLOBAL BUS STOP	44
29	ESTUDIANTE CUBANO	18
30	SMOKING ARCHITECT	28
32	STATE VISIT	17
33	EVENEMENT	17
34	LE TOUT MIS EN PERSPECTIVE	15
36	WITHOUT QUESTION	17
37	ãLEMENTA PICTURã	12

tag

id	tag	numrel	datains
1	profeta	1	NULL
2	precetti	1	NULL
3	nome	2	NULL
4	verbo	1	NULL
5	parola	4	NULL
6	padre	5	NULL
7	fondamentalismo	2	NULL
8	guida	4	NULL
9	islam	5	NULL
10	pace	23	NULL

```
tagopera
+------+-----------+---------+-------+
| id   | id0_opera | id1_tag | peso  |
+------+-----------+---------+-------+
| 4038 |        10 |     709 | 1.100 |
| 4037 |        10 |      11 | 1.000 |
| 4036 |        10 |      10 | 1.000 |
| 4035 |        10 |       9 | 1.000 |
| 4034 |        10 |       8 | 1.000 |
| 4033 |        10 |       1 | 1.000 |
| 4032 |        10 |       6 | 1.000 |
| 4031 |        10 |       7 | 1.000 |
| 4030 |        10 |       2 | 1.000 |
| 4029 |        10 |       3 | 1.271 |
+------+-----------+---------+-------+
```

Dalle descrizioni delle precedenti tre tabelle si vede come sia i tag che le opere siano individuate da un intero; questo intero è univoco nello spazio dei tag e univoco nello spazio delle opere; l'unicità non è più garantita quando si consideri l'unione; infatti, la query

```
select * from tagopera where id0_opera = id1_tag;
```

```
+-------+-----------+---------+-------+
| id    | id0_opera | id1_tag | peso  |
+-------+-----------+---------+-------+
|  4036 |        10 |      10 | 1.000 |
|  4576 |       326 |     326 | 1.000 |
| 16590 |      1246 |    1246 | 1.000 |
| 17141 |       723 |     723 | 1.000 |
| 29206 |       704 |     704 | 1.000 |
+-------+-----------+---------+-------+
```

mostra come esistano opere e tag che sono individuate dallo stesso intero; quindi un vettore pesato risultato (ottenuto dalla tabella precedente) potrebbe essere il seguente (i valori riportati non sono ovviamente i valori "finali" in quanto la loro somma dovrebbe essere 1, ma sono stati usati semplicemente come riempitivo):

W[K] = (... 1.0, 1.0, 1.0, 1.0, 1.0, ...)

che corrispondono alle risorse R (opere o tag)

R[K] = (... 10, 326, 1246, 723, 704, ...)

(vedere colonne id0_opera e id1_tag della tabella precedente).

Ma a questo punto, come si fa a sapere, per esempio, se la risorsa con id 326 si riferisce a id0_opera o a id1_tag? Non si può sapere, a meno che non si usi la convenzione che tutte le opere precedono i tag: in questo caso, contando quante sono le opere, si sa che da un certo indice in poi si tratta di tag.

Comincia a trasparire la necessità di associare un indice che sia univoco nello spazio,più ampio, dell'unione di opere e tag (spazio che chiamo spaziodelle "risorse", dove per risorsa si intende o una opera o un tag): il termine"unione" qui è utilizzato in modo un pò improprio, perché, per definizione diunione insiemistica, gli elementi uguali verrebbero considerati una volta sola (serve che ENTRAMBI siano considerati, perchè con significato diverso). Una volta associato un indice univoco, non sembrerebbe più importante l'ordinamento (tuttavia un ordinamento compare ugualmente: vedere nel seguito).

Ecco le query un pò più da vicino:

1) Si comincia con l'ordinare le opere per id0_opera crescente e, a parità di id0_opera si ordina per id1_tag crescente (il motivo dell'ordinamento sarà chiaro fra breve)

2) Data un id0_opera, si calcola la somma dei pesi degli archi uscenti da quella id0_opera; questo si fa per tutte le opere.

Ecco le due query, come sempre limitate a poche righe (chiamo le tabelle ottenute rispettivamente Q1 e Q2)

1) select * from tagopera order by id0_opera, id1_tag;

2) select id0_opera, sum(peso) as pesotot from tagopera group by id0_opera order by id0_opera;

1) Q1

id	id0_opera	id1_tag	peso
4033	10	1	1.000
4030	10	2	1.000
4029	10	3	1.271
4028	10	4	1.000
4027	10	5	1.000
4032	10	6	1.000
4031	10	7	1.000
4034	10	8	1.000
4035	10	9	1.000
4036	10	10	1.000

2) Q2

id0_opera	pesotot
10	12.371
27	12.000
28	11.000
29	13.100
30	18.000
32	12.000
33	12.000
34	15.000
36	14.000
37	14.000

Q1 evidenzia che dall'opera 10 esce un arco verso il tag 1 con peso 1.000; poi che esce un altro arco verso il tag 2 con peso 1.000; poi ancora un altro arco verso il tag 3 con peso 1.271; e così via,... mentre Q2 evidenzia che la somma di tutti gli archi uscenti dall'opera 10 è pari a 12.371... In questo modo si può far calcolare da mysql il valore che verrà messo in una certa posizione della matrice H; infatti, nella posizione della matrice H relativa per esempio all'opera 10 e al tag 1 (di cui non abbiamo ancora calcolato gli indici i,j) andrà un valore pari al peso del link fra opera 10 e tag 1 (che vale 1.000) DIVISO per il peso totale degli archi uscenti dall'opera 10 (che è pari a 12.371): questo calcolo si fa con la seguente query (utilizzando le Q1 e Q2):

3) select Q1.id0_opera, Q1.id1_tag, Q1.peso/Q2.pesotot as pesodx from Q1, Q2 where Q1.id0_opera = Q2.id0_opera;

Si ottiene la tabella che chiamo DX;

DX

id0_opera	id1_tag	pesodx
10	1	0.0808342
10	2	0.0808342
10	3	0.1027403
10	4	0.0808342
10	5	0.0808342
10	6	0.0808342
10	7	0.0808342
10	8	0.0808342
10	9	0.0808342
10	10	0.0808342

osservando le righe, il significato è il seguente: nella matrice H, in riga 10 e colonna 1, inserisci il valore 0.0808342. Attenzione: a questo punto, non è stata fatta ancora nessuna associazione indice->risorsa; quindi il significato esatto è il seguente: nella matrice H, in riga i(10) e colonna j(1), inserisci il valore 0.0808342 dove i(10) e j(1) saranno determinati facendo uso di un'altra tabella.

La successiva query fa uso di una tabella di supporto (che è la tabella in cui è contenuta l'associazione indice->risorsa e che è stata creata con una query che sarà trattata verso la fine di questa descrizione); osservando la tabella di supporto si vede perché è necessario l'ordinamento (intendendolo nel senso di mettere prima tutte le opere poi tutti i tag, o viceversa): infatti nonostante gli indici, la colonna risorsa non ci dice se "risorsa" è una opera o un tag.

Ecco la tabella di supporto:

SUP

indice	risorsa
0	10
1	27
2	28
3	29
4	30
5	32
6	33
7	34
8	36
9	37

Considerando le tabelle DX e SUP, l'interrogazione

(where sup.risorsa = dx.id0_opera and indice < #opere)

ci dice che una risorsa presa da sup è una opera quando tale risorsa è presa dalla tabella DX E contemporaneamente ha INDICE strettamente minore del numero di opere (strettamente minore perchè le opere vanno da 0 a #opere - 1). Si noti ancora una volta che in SUP è associato un indice a una risorsa; ad es, non si capisce se la risorsa 10 è una opera o un tag (si ricordi che esiste sia una opera con id 10, sia un tag con id 10); qui non si riesce ancora a capire chi sia 10.

Ordinando prima le opere e contandole e poi i tag (o viceversa, prima i tag e poi le opere), si è a conoscenza del fatto che le prime N risorse sono opere e le restanti sono tag (o viceversa). Altrimenti non si ha modo di distinguere opere e tag.

La seguente query, utilizzando la tabella di supporto, è quella che permette di indicare le righe della matrice H in cui mettere un valore ed è ristretta alle sole opere:

select indice as i, id1_tag, dx.pesodx from sup, dx where sup.risorsa = dx.id0_opera and indice < (select count(id) from opera);

Attenzione: la parte di query "indice < (select count(id) from opera);", indica che sto considerando solo gli indici relativi alle opere; infatti si sta riempiendo la parte superiore destra della matrice H (si è scelto di mettere prima tutte le opere seguite dai tag: la fig.9.3 rende più chiara la situazione); un'altro gruppo di query come queste sarà utilizzato per riempire la parte inferiore sinistra di H.

Nella tab ottenuta dall'ultima query, riportata sotto e chiamata DESTRA, ci sono gli indici "i" nel formato giusto, mentre non ci sono ancora gli indici "j" (si intende indici i,j di H). I valori della colonna id1_tag della tabella "DESTRA" devono essere convertiti in indici di colonna della matrice H; per far questo bisogna completare la query considerando le risorse che sono tag (prossima query).

Questa tabella la chiamo DESTRA ad indicare che si intende riempire la matrice H considerando i legami da opera a tag (e quindi si procede camminando da sinistra verso DESTRA nel grafo rappresentante le relazioni opera->tag).

DESTRA

i	id1_tag	pesodx
0	1	0.0808342
0	2	0.0808342
0	3	0.1027403
0	4	0.0808342
0	5	0.0808342
0	6	0.0808342
0	7	0.0808342
0	8	0.0808342
0	9	0.0808342
0	10	0.0808342

Consideriamo adesso anche i tag:

select i, indice as j, pesodx from sup, destra where sup.risorsa = destra.id1_tag and indice > (select (count(id) from opera);

La tabella che otteniamo è la seguente, dove non sono state considerate le prime righe, ma delle righe che si trovano più o meno nel mezzo.

i	j	pesodx
2764	2998	0.1666667
2764	4687	0.1666667
2764	7167	0.1666667
2764	9447	0.1666667
2764	13204	0.1666667
11	20566	0.0471743
18	20641	0.0714286
28	18645	0.0709220
30	18645	0.0406009
55	18631	0.0632151

Una cosa che non si notava immediatamente è il mancato ordinamento per "i", nel senso che valori più piccoli di i possono seguire valori più grandi, come si vede dal pezzo di tabella riportato. Non è molto importante questo ordinamento, perché la matrice viene costruita correttamente. Tuttavia per il modo in cui sono stati costruiti alcuni metodi della classe MatriceSparsa è meglio mantenere l'ordinamento con tutti gli indici disposti in ordine crescente (il motivo è spiegato più avanti, nel metodo "sottostocastica" della classe MatriceSparsa).

Una aggiunta finale quindi: si richiede di ordinare rispetto agli indici:

select i, indice as j, pesodx from sup, destra where sup.risorsa = destra.id1_tag and indice > (select count(id) from opera) order by i, j; In maniera analoga si procede per costruire la parte inferiore sinistra della matrice H (le due query complete si trovano in ArsConn.getH()).

9.3.3.4 La classe

```
import java.sql.*; // Per l'accesso al DB
import java.util.Scanner; // Per alcuni input

//AVVERTENZA
// Se il programma viene fatto eseguire su  una macchina diversa da quella che ospita
// il database, gli Statement sono da cambiare in PreparedStatement per prevenire
// attacchi di tipo SqlInjection

public class ArsConn {

  // Parametro di connessione al database
  private static String connStr;

  // Oggetto per la connessione
  private static Connection conn;

  // Variabili per il calcolo dei termini del FolkRank
  private static int number_of_operas;
  private static int number_of_tags;
  private static int dim; // dimensione n di H(n,n)
  private static boolean check;
  private static boolean drop_sup;
  private static boolean create_sup;
  private static int cursor; // uso: scorrimento indici/righe/posizioni
```

```
private static boolean fill_sup;
private static int dimRapp; // dimensione rappresentazione di H
private static int dimRappA; // dimensione rappresentazione di A

private MatriceSparsa ArsMatrix; // matrice H
private MatriceSparsa ArsVector; // vettore di partenza
private int[] ArsHanger; // vettore delle pagine pendenti

/**
```

Nel costruttore si definisce la stringa di connessione e il suo valore e in seguito si fanno alcuni controlli sul database (vedere il metodo checkDB()).

Se i controlli danno esito negativo, il programma non viene eseguito.

Se i controlli danno esito positivo, il costruttore prosegue con la inizializzazione di quelle variabili utili al calcolo del FolkRank.

```
*/
public ArsConn() {

    // Nell'ordine in cui sono:
    connStr = dfConnStr();
    conn = null;
    check = checkDB(); // controllo di consistenza

    if (check) {

      number_of_operas = getNumOperas();
      number_of_tags = getNumTags();
      dim = number_of_operas + number_of_tags;
      drop_sup = dropSup();
      create_sup = createSup();
      cursor = 0;
      fill_sup = fillSup();
      dimRapp = getDimRapp();
```

```
      ArsMatrix = getH();
      dimRappA = getDimRappA();
      ArsHanger = getPendenti();
      ArsVector =  getVet1();

    }

    else {
      System.out.println("STOP");
    }

  } // fine costruttore


   /**
      Passaggio di parametri di connessione, attraverso il metodo
DriverManager.getConnection(String connectionString)
      La forma della connectionString è la seguente
    jdbc:mysql://[H][:P]/[db][?param][=valore][&param2][=valore2]
 dove H è il nome o l'indirizzo IP e P (= porta) è generalmente la 3306.
      Si comincia coll'impostare la connectionString.
      Il metodo chiede di immettere alcune informazioni.
      @return connStr, la stringa di connessione.
   */
   public static String dfConnStr() {

    Scanner sc = new Scanner(System.in);
    String str = "";

    System.out.println();

    String host = "";
```

```
 System.out.print(
   "HOST (nome o ip) su cui esegue MySql (\"localhost\"): ");
 host = sc.next();

 String port = "";
 System.out.print(
   "PORTA su cui e' in ascolto MySql (3306): ");
 port = sc.next();

 String dbname = "";
 System.out.print(
   "NOME_DB MySql: ");
 dbname = sc.next();

 String user = "";
 System.out.print(
   "UTENTE MySql (garantire SELECT, INSERT, UPDATE, CREATE, DROP): ");
 user = sc.next();

    String pwd = PasswordApp.returnPassword();

    System.out.println();

 str =
   "jdbc:mysql://" + host + ":" + port + "/" + dbname + "?user=" +
   user + "&password=" + pwd;

 return str;

}

/**
```

Si connette al database, esegue una query e restituisce un int rappresentante il numero di opere trovate nel database.

```
    @return num_of_operas, il numero delle opere presenti nel db.
   */
   public static int getNumOperas() {
    System.out.println("getNumOperas()");
    conn = null;
    String statement_number_of_operas = "select count(id) from opera;";
    int num_of_operas = 0;
    try {
     Class.forName("com.mysql.jdbc.Driver");
     conn = DriverManager.getConnection(connStr);
     Statement stmt_number_of_operas = conn.createStatement();
     ResultSet rs_number_of_operas =
      stmt_number_of_operas.executeQuery(
       statement_number_of_operas);
     while(rs_number_of_operas.next())
      num_of_operas = Integer.parseInt(
       rs_number_of_operas.getString(1));
     conn.close();
    }
    catch(ClassNotFoundException e) { e.printStackTrace(); }
    catch(SQLException e) { e.printStackTrace(); }
    catch(Exception e) { e.printStackTrace(); }
    finally { return num_of_operas; }
   }

   /**
       Controllo "di consistenza"
       Vi sono alcune situazioni che possono disturbare l'esecuzione del programma
      già fin dall'inizio (in particolare consistenze che non devono esistere se la
      base di dati è ben progettata).
```

```
     In caso di inconsistenze, il programma non prosegue nell'elaborazione.
    @return res, un valore di verità (se res = false il programma viene interrotto,
     se res = true si prosegue).
*/
public static boolean checkDB() {
  System.out.println("checkDB()\n");
  boolean res = false;
  conn = null;

  String statement_po =
     "select count(id) from opera where id not in (select id0_opera from tagopera);";
  String statement_io =
    "select count(id0_opera) from tagopera where id0_opera not in " +
   "(select id from opera);";
  String statement_pt =
    "select count(id) from tag where id not in (select id1_tag from tagopera);";
  String statement_it =
    "select count(id1_tag) from tagopera where id1_tag not in (select id from tag);";

  int po, pt, io, it; // opere o tag pendenti e opere o tag inconsistenti
  po = pt = io = it = -1;

  try {
   Class.forName("com.mysql.jdbc.Driver");
   conn = DriverManager.getConnection(connStr);

   Statement stmt_po = conn.createStatement();
   ResultSet rs_po = stmt_po.executeQuery(statement_po);
   while(rs_po.next()) po = Integer.parseInt(rs_po.getString(1));
   System.out.println("Opere pendenti = " + po);

   Statement stmt_pt = conn.createStatement();
```

```
    ResultSet rs_pt = stmt_pt.executeQuery(statement_pt);
    while(rs_pt.next()) pt = Integer.parseInt(rs_pt.getString(1));
    System.out.println("Tag pendenti = " + pt);

    Statement stmt_io = conn.createStatement();
    ResultSet rs_io = stmt_io.executeQuery(statement_io);
    while(rs_io.next()) io = Integer.parseInt(rs_io.getString(1));
    System.out.println("Opere inconsistenti = " + io);

    Statement stmt_it = conn.createStatement();
    ResultSet rs_it = stmt_it.executeQuery(statement_it);
    while(rs_it.next()) it = Integer.parseInt(rs_it.getString(1));
    System.out.println("Tag inconsistenti = " + it + "\n");

    if (io > 0 || it > 0) res = false;
    else res = true;

    conn.close();
  }
  catch(ClassNotFoundException e) { e.printStackTrace(); }
  catch(SQLException e) { e.printStackTrace(); }
  catch(Exception e) { e.printStackTrace(); }
  finally { return res; }
}

public static boolean getCheck() { return check; }

/**
    Si connette al database, esegue una query e restituisce un int
  rappresentante il numero di tag trovate nel database.
  @return int num_of_tags, il numero dei tag presenti nel db.
*/
```

```
public static int getNumTags() {
  System.out.println("getNumTags()");
  conn = null;
  String statement_number_of_tags = "select count(id) from tag;";
  int num_of_tags = 0;
  try {
    Class.forName("com.mysql.jdbc.Driver");
    conn = DriverManager.getConnection(connStr);
    Statement stmt_number_of_tags = conn.createStatement();
    ResultSet rs_number_of_tags =
      stmt_number_of_tags.executeQuery(
        statement_number_of_tags);
    while(rs_number_of_tags.next())
      num_of_tags = Integer.parseInt(
        rs_number_of_tags.getString(1));
    conn.close();
  }
  catch(ClassNotFoundException e) { e.printStackTrace(); }
  catch(SQLException e) { e.printStackTrace(); }
  catch(Exception e) { e.printStackTrace(); }
  finally { return num_of_tags; }
}

/**
    Se si esegue un nuovo calcolo, i precedenti valori di FolkRank
  non servono più (perché sono diventati vecchi o perché sono state
  introdotte nuove opere e/o tag nel database); così questo metodo
  cancella (se esiste) la tabella sup prima di cominciare con un
  nuovo calcolo
  @return rs_drop_sup, un valore di verita' (true se l'operazione ha
    avuto successo,false altrimenti).
*/
```

```
public static boolean dropSup() {
  System.out.println("dropSup()");
  conn = null;
  boolean rs_drop_sup = false;
  String statement_drop_sup = "drop table if exists sup;";
  try {
    Class.forName("com.mysql.jdbc.Driver");
    conn = DriverManager.getConnection(connStr);
    Statement stmt_drop_sup = conn.createStatement();
    rs_drop_sup =
      stmt_drop_sup.execute(statement_drop_sup);
    conn.close();
  }
  catch(ClassNotFoundException e) { e.printStackTrace(); }
  catch(SQLException e) { e.printStackTrace(); }
  catch(Exception e) { e.printStackTrace(); }
  finally { return rs_drop_sup; }
}

/**
    Nel database è necessaria una tabella di supporto in cui siano
  presenti le associazioni indice -> risorsa, dove indice è una
  colonna del vettore risultato e risorsa è l'opera (o il tag)
  associata a quell'indice.
    Viene perciò creata fisicamente (su disco) una nuova tabella in
  aggiunta a quelle già presenti nel database.
  @return rs_create_sup, un valore di verita' (true se l'operazione ha
    avuto successo, false altrimenti).
*/
public static boolean createSup() {
  System.out.println("createSup()");
  conn = null;
```

```
  boolean rs_create_sup = false;
  String statement_create_sup =
    "create table sup (" +
    "indice int(11) primary key, " +
    "risorsa int(11), " +
    "fr decimal(24, 21));";
  try {
    Class.forName("com.mysql.jdbc.Driver");
    conn = DriverManager.getConnection(connStr);
    Statement stmt_create_sup = conn.createStatement();
    rs_create_sup =
      stmt_create_sup.execute(statement_create_sup);
    conn.close();
  }
  catch(ClassNotFoundException e) { e.printStackTrace(); }
  catch(SQLException e) { e.printStackTrace(); }
  catch(Exception e) { e.printStackTrace(); }
  finally { return rs_create_sup; }
}

/**
    Riempie la tabella di supporto.
    Prima di iniziare il calcolo del FolkRanK, non si conoscono i
   valori del vettore finale; per questo nella colonna fr
   viene inserito un valore fittizio (-1, uguale per tutte le risorse)
   @return rs_fill_sup, un valore di verita' (true se l'operazione ha
    avuto successo,false altrimenti)
*/
public static boolean fillSup() {
  System.out.println("fillSup()");
  //div_prec = setDivPrec();
  conn = null;
```

```
cursor = 0;
boolean rs_fill_sup = false;
String statement_risorsa_opera =
  "select id from opera order by id;";
String statement_risorsa_tag =
  "select id from tag order by id;";
try {
 Class.forName("com.mysql.jdbc.Driver");
 conn = DriverManager.getConnection(connStr);
 Statement stmt_risorsa_opera = conn.createStatement();
 Statement stmt_risorsa_tag = conn.createStatement();
 ResultSet rs_risorsa_opera =
  stmt_risorsa_opera.executeQuery(statement_risorsa_opera);
 ResultSet rs_risorsa_tag =
  stmt_risorsa_tag.executeQuery(statement_risorsa_tag);
 // riempie un numero di righe corrispondenti al numero di opere
 while(rs_risorsa_opera.next()) {
  String risorsa =
   "\"" + rs_risorsa_opera.getString(1) + "\"";
  String statement_riempi =
   "insert into sup values(" +
   cursor++ + ", " +
   risorsa + ", " +
   -1 + ");";
  Statement stmt_riempi = conn.createStatement();
  stmt_riempi.executeUpdate(statement_riempi);
 }
 // riempie le restanti righe (in numero pari al numero di tag)
 while(rs_risorsa_tag.next()) {
  String risorsa =
   "\"" + rs_risorsa_tag.getString(1) + "\"";
  String statement_riempi =
```

```
        "insert into sup values(" +
        cursor++ + ", " +
        risorsa + ", " +
        -1 + ");";
      Statement stmt_riempi = conn.createStatement();
      stmt_riempi.executeUpdate(statement_riempi);
    }
    conn.close();
    // se l'esecuzione è arrivata a questo punto, ok
    rs_fill_sup = true;
  }
  // se l'esecuzione si ferma in uno di questi, rs_fill_sup = false
  catch(ClassNotFoundException e) { rs_fill_sup = true; e.printStackTrace(); }
  catch(SQLException e) { rs_fill_sup = true; e.printStackTrace(); }
  catch(Exception e) { rs_fill_sup = true; e.printStackTrace(); }
  finally { return rs_fill_sup; }
}

/**
    La matrice sottostocastica che compare nel calcolo del FolkRank
  è una matrice sparsa (la matrice H); ciò significa che è possibile
  rappresentare solamente gli elementi che sono diversi dall'elemento
  dominante, con grande risparmio di memoria. Per far questo, però, è
  necessario conoscere quanti sono gli elementi da inserire in H, in
  modo da dare una dimensione al vettore che rappresenterà la H.
  @return sizeH, la dimensione (di tipo int) della rappresentazione di H.
*/
 public static int getDimRapp() {
  System.out.println("getDimRapp()");
  conn = null;
  int sizeH = 0;
  String statement_dim_rapp = "select count(*) from tagopera;";
```

```
 try {
  Class.forName("com.mysql.jdbc.Driver");
  conn = DriverManager.getConnection(connStr);
  Statement stmt_dim_rapp = conn.createStatement();
  ResultSet rs_dim_rapp =
   stmt_dim_rapp.executeQuery(statement_dim_rapp);
  while(rs_dim_rapp.next())
   // qui bisogna moltiplicare per due perché bisogna tenere conto
   // del #archi da opere a tag
   // del #archi da tag a opere
   // siccome sono archi non orientati, ogni arco vale per due
   sizeH = 2 * Integer.parseInt(rs_dim_rapp.getString(1));
  conn.close();
 }
 catch(ClassNotFoundException e) { e.printStackTrace(); }
 catch(SQLException e) { e.printStackTrace(); }
 catch(Exception e) { e.printStackTrace(); }
 finally { return sizeH; }
}

/**
   Costruisce la matrice H.
  Prima costruisce la parte superiore destra di H (equivale a
 considerare il grafo delle relazioni percorrendolo da opera a tag).
  Poi costruisce la parte inferiore sinistra di H (equivale e
 considerare il grafo delle relazioni percorrendolo da tag a opera).
  La matrice restituita è sottostocastica.
  In mysql l'impostazione predefinita dell'operatore di divisione
 è insufficiente per ottenere una precisione maggiore di 10E-7: è
 per questo che è necessario indicare nel file di configurazione di
 produrre un numero magdiore di cifre decimali.
 @return msH, la matrice H (di tipo MatriceSparsa)
```

```
   */
   public static MatriceSparsa getH() {
    System.out.println("getH()");
    ElementoMatriceSparsa ems;
    ElementoMatriceSparsa[] ems_mat =
     new ElementoMatriceSparsa[dimRapp];
    MatriceSparsa msH = new MatriceSparsa();
    cursor = 0;
    conn = null;
    String statement1 =
     "select i, indice as j, pesodx from " +
      "sup, " +
      "(select indice as i, id1_tag, dx.pesodx from " +
       "sup, " +
       "(select Q1.id0_opera, Q1.id1_tag, Q1.peso/Q2.pesotot as pesodx from " +
        "(select * from tagopera order by id0_opera, id1_tag) as Q1, " +
        "(select id0_opera, sum(peso) as pesotot from tagopera group by id0_opera
order by id0_opera) as Q2 " +
        "where Q1.id0_opera = Q2.id0_opera) as dx " +
       "where sup.risorsa = dx.id0_opera and indice < (select count(id) from opera) ) as
destra " +
      "where sup.risorsa = destra.id1_tag and indice > (select count(id) from opera)-1 " +
      "order by i, j;";
    String statement2 =
     "select i, indice as j, pesosx from " +
      "sup, " +
      "(select indice as i, id0_opera, sx.pesosx from " +
       "sup, " +
       "(select Q1.id0_opera, Q1.id1_tag, Q1.peso/Q2.pesotot as pesosx from " +
        "(select * from tagopera order by id1_tag, id0_opera) as Q1, " +
        "(select id1_tag, sum(peso) as pesotot from tagopera group by id1_tag order by
id1_tag) as Q2 " +
```

```
          "where Q1.id1_tag = Q2.id1_tag) as sx " +
          "where sup.risorsa = sx.id1_tag and indice > (select count(id) from opera)-1 ) as
sinistra " +
       "where sup.risorsa = sinistra.id0_opera and indice < (select count(id) from opera) "
+
       "order by i, j;";
     try {
      Class.forName("com.mysql.jdbc.Driver");
      conn = DriverManager.getConnection(connStr);
      Statement stmt1 = conn.createStatement();
      ResultSet rs1 = stmt1.executeQuery(statement1);
      // prima parte della matrice
      while(rs1.next()) {
        int indiceRiga = Integer.parseInt(rs1.getString(1));
        int indiceColonna = Integer.parseInt(rs1.getString(2));
        double pesoDestro = Double.parseDouble(rs1.getString(3));
        ems = new ElementoMatriceSparsa(
          indiceRiga, indiceColonna, pesoDestro);
        ems_mat[cursor++] = ems;
      }
      // seconda parte della matrice
      Statement stmt2 = conn.createStatement();
      ResultSet rs2 = stmt1.executeQuery(statement2);
      while(rs2.next()) {
        int indiceRiga = Integer.parseInt(rs2.getString(1));
        int indiceColonna = Integer.parseInt(rs2.getString(2));
        double pesoSinistro = Double.parseDouble(rs2.getString(3));
        ems = new ElementoMatriceSparsa(
          indiceRiga, indiceColonna, pesoSinistro);
        ems_mat[cursor++] = ems;
      }
      msH = new MatriceSparsa(ems_mat, dim, dim, dimRapp);
```

```
    conn.close();
   }
   catch(ClassNotFoundException e) { e.printStackTrace(); }
   catch(SQLException e) { e.printStackTrace(); }
   catch(Exception e) { e.printStackTrace(); }
   finally { return msH; }
}

/**
    La matrice A è quella che risolve il problema delle pagine
   pendenti.
    Opere che non hanno nessuna relazione con tag e tag che non
   hanno nessuna relazione con alcuna opera, producono pagine
   pendenti. Questo metodo identifica le risorse che hanno queste
   caratteristiche.
    Questo metodo calcola la dimensione della rappresentazione
   di A.
   @return dimA, la dimensione della rappresentazione di A
*/
public static int getDimRappA() {
   System.out.println("getDimRappA()");
   //cursor = 0;
   int dimA = 0;
   conn = null;
   /*
      Breve descrizione delle seguenti query.
       La query interna seleziona tutte le id dei tag presenti
     nella tabella tagopera.
      La query esterna conta tutti i tag che non sono in
     tagopera.
      Esclude tutte le relazioni fra tag e opera aventi numrel=0.
      Esclede anche le inconsistenze note (cioè quelle per cui
```

```
    esistono relazioni fra tag e opera aventi numrel != 0 MA
    non contenute in tagopera)
  */
  String statement4 =
    // seleziona tutte le tag.id NON in tagopera(.id1_tag) e le conta
    "select count(id) from tag where id not in (select id1_tag from tagopera);";
  String statement44 =
    // seleziona tutte le opera.id NON in tagopera(.id0_opera) e le conta
    "select count(id) from opera where id not in (select id0_opera from tagopera);";

  try {
    Class.forName("com.mysql.jdbc.Driver");
    conn = DriverManager.getConnection(connStr);
    Statement stmt4 = conn.createStatement();
    ResultSet rs4 = stmt4.executeQuery(statement4);
    while(rs4.next()) dimA = Integer.parseInt(rs4.getString(1));
    Statement stmt44 = conn.createStatement();
    ResultSet rs44 = stmt44.executeQuery(statement44);
    while(rs44.next()) dimA += Integer.parseInt(rs44.getString(1));
    conn.close();
  }
  catch(ClassNotFoundException e) { e.printStackTrace(); }
  catch(SQLException e) { e.printStackTrace(); }
  catch(Exception e) { e.printStackTrace(); }
  finally { return dimA; }
}

/**
    Questo metodo costruisce e restituisce un vettore che contiene
  alcuni indici.
    Tali indici corrispondono alle righe zero di H.
  @return pendenti, un vettore di interi
```

```
   */
   public static int[] getPendenti() {
    // Controlla tag e opere pendenti
    System.out.println("getPendenti()");
    int[] pendenti = new int[dimRappA];
    conn = null;
    cursor = 0;
    // Seleziona tutte le tag aventi numrel = 0 (qs tag danno origine a righe pendenti)
    String statement3 =
    "select sup.indice from " +
      "(select * from tag where id not in (select id1_tag from tagopera)) as t, sup " +
       "where t.id = sup.risorsa and indice > (select count(id) from opera)-1 order by sup.indice;";
    String statement33 =
    "select sup.indice from " +
      "(select * from opera where id not in (select id0_opera from tagopera)) as t, sup " +
       "where t.id = sup.risorsa and indice < (select count(id) from opera) order by sup.indice;";
    try {
     Class.forName("com.mysql.jdbc.Driver");
     conn = DriverManager.getConnection(connStr);
      // Metto prima tutte le opere pendenti (se ce ne sono): nella tabella sup vi sono prima le opere e poi i tag
     Statement stmt33 = conn.createStatement();
     ResultSet rs33 = stmt33.executeQuery(statement33);
     while(rs33.next()) {
      int indiceRigaPendente = Integer.parseInt(rs33.getString(1));
      pendenti[cursor++] = indiceRigaPendente;
     }
     // Metto poi tutti i tag pendenti (se ce ne sono)
     Statement stmt3 = conn.createStatement();
     ResultSet rs3 = stmt3.executeQuery(statement3);
```

```
    while(rs3.next()) {
      int indiceRigaPendente = Integer.parseInt(rs3.getString(1));
      pendenti[cursor++] = indiceRigaPendente;
    }
    conn.close();
  }
  catch(ClassNotFoundException e) { e.printStackTrace(); }
  catch(SQLException e) { e.printStackTrace(); }
  catch(Exception e) { e.printStackTrace(); }
  finally { return pendenti; }
}

/**
    Questo metodo restituisce un vettore iniziale
  adattato ad arsmeteo.
  @return MS_vet, il vettore iniziale per arsmeteo
*/
public static MatriceSparsa getVet1() {
  System.out.println("getVet1()");
  conn = null;
  cursor = 0;
  ElementoMatriceSparsa ems;
  ElementoMatriceSparsa[] ems_vet = new ElementoMatriceSparsa[dim];
  MatriceSparsa vet = new MatriceSparsa();
  String ultime26opere =
  "select indice from " +
    "(select * from " +
      "(select * from opera order by id desc limit 26) as 26op_reversed " +
      "order by id asc) as 26op, " +
    "sup " +
    "where 26op.id = sup.risorsa and " +
    "sup.indice < (select count(*) from opera) order by indice;";
```

```
String prime50tag =
 "select indice from (select id1_tag, sum(peso) as pesotot from tagopera " +
 "group by id1_tag order by pesotot desc limit 50) as 50t, sup where 50t.id1_tag = "
+
"sup.risorsa and sup.indice > (select count(*) from opera) - 1 " +
"order by indice;";
try {
 Class.forName("com.mysql.jdbc.Driver");
 conn = DriverManager.getConnection(connStr);
 Statement stmt_ultime26opere = conn.createStatement();
 ResultSet rs_ultime26opere = stmt_ultime26opere.executeQuery(ultime26opere);
 Statement stmt_prime50tag = conn.createStatement();
 ResultSet rs_prime50tag = stmt_prime50tag.executeQuery(prime50tag);
 // Nei while:
 // Considero 3 tipi di risorse:
 // - le ultime 26 opere
 // - le "prime" 50 tag
 // - le rimanenti risorse (a cui sono state tolte le 26op e le 50t)
 // Allora questi sono i valori per il db:
 // 78 = 3 * 26
 // 150 = 3 * 50
 // X = 3 * RIMANENTI
 int opere = 0; int tag = 0; // servono per contare opere e tag
 int rimanenti = (number_of_operas + number_of_tags) - 50 - 26;
 while(rs_ultime26opere.next()) {
  int indiceOperaCorrente = Integer.parseInt(rs_ultime26opere.getString(1));
  opere++;
  while (cursor < indiceOperaCorrente) {
   ems = new ElementoMatriceSparsa(0, cursor, 1.0 / (3 * rimanenti));
   ems_vet[cursor++] = ems;
  }
  double den = 1.0 / 78;
```

```
        ems = new ElementoMatriceSparsa(0, cursor, den);
        ems_vet[cursor++] = ems;
      }
      while(rs_prime50tag.next()) {
        int indiceTagCorrente = Integer.parseInt(rs_prime50tag.getString(1));
        tag++;
        while (cursor < indiceTagCorrente) {
          ems = new ElementoMatriceSparsa(0, cursor, 1.0 / (3 * rimanenti));
          ems_vet[cursor++] = ems;
        }
        double den = 1.0 / 150;
        ems = new ElementoMatriceSparsa(0, cursor, den);
        ems_vet[cursor++] = ems;
      }
      // potrebbero esserci altri valori
      while (cursor < dim) {
        ems = new ElementoMatriceSparsa(0, cursor, 1.0 / (3 * rimanenti));
        ems_vet[cursor++] = ems;
      }
      vet = new MatriceSparsa(ems_vet, 1, dim, dim);

      System.out.println("\n" + opere + "/26 opere considerate per la personalizzazione del vettore iniziale");
      System.out.println(tag + "/50 tag considerati per la personalizzazione del vettore iniziale\n");

      if (opere < 26 || tag < 50)
        System.out.println("Opere o tag insufficienti per inizializzare correttamente il vettore iniziale");

      conn.close();
    }
```

```
 catch(ClassNotFoundException e) { e.printStackTrace(); }
 catch(SQLException e) { e.printStackTrace(); }
 catch(Exception e) { e.printStackTrace(); }
 finally { return vet; }
}

/**
   Un metodo per mettere i risultati (i rank calcolati)
 nella terza colonna della tabella di supporto.
 @param vS il vettore risultato
*/
public static void updateSup(MatriceSparsa vS) {
 System.out.println("updateSup()");
 Connection conn = null;
 try {
  Class.forName("com.mysql.jdbc.Driver");
   conn = DriverManager.getConnection(connStr);
   for(int index = 0; index < vS.getNumColonne(); index++){
   double rank = vS.getMatriceSparsa()[index].restituisciValore();
   String statement_aggiorna =
     "update sup set fr = " + rank + " where indice = " + index + ";";
   Statement stmt_aggiorna = conn.createStatement();
   stmt_aggiorna.executeUpdate(statement_aggiorna);
  }
  conn.close();
 }
 catch(ClassNotFoundException e) { e.printStackTrace(); }
 catch(SQLException e) { e.printStackTrace(); }
 catch(Exception e) { e.printStackTrace(); }
 finally { System.out.println(); }
}
```

```
    /**
       Restituisce la matrice H
       @return ArsMatrix, la matrice H
    */
    public MatriceSparsa getMatrix() { return ArsMatrix; }

    /**
       Restituisce il vettore (iniziale);
       @return ArsVector, il vettore W0
    */
    public MatriceSparsa getVector() { return ArsVector; }

    /**
       Restituisce il vettore delle pagine pendenti
       NOTA: per limitare il consumo di memoria, la matrice A è
      rappresentata da un vettore contenente gli indici delle righe
      zero.
       Eventuali operazioni che coinvolgono il vettore delle pagine
      pendenti sono regolate dalla classe MatriceSparsa.
      @return ArsHanger, il vettore delle pagine pendenti
    */
    public int[] getHanger() { return ArsHanger; }
}
```

9.3.4 La classe FR

```
/**
```

Questa classe crea un oggetto sul quale vengono applicati i metodidelle classi precedenti, per poter calcolare i valori di FolkRank mediante il metodo delle potenze.

```
*/
public class FR{

        private MatriceSparsa pi_NEXT; //array dei FR all'iterazione corrente
```

```
private MatriceSparsa pi_PREV; //array dei FR alla frecedente iterazione
private MatriceSparsa pi_ZERO; //array dei FR iniziale
private static final double ALFA = 0.85; // surf regolare o pagine pendenti
private static MatriceSparsa H; // matrice delle relazioni (da tagopera)
private static int[] a; // pagine pendenti: ARRAY con elementi tutti uguali
private static final double CO_ALFA = 0.15; // surf irregolare per "noia"
private static final double EPSILON = java.lang.Math.pow(10, -9);

public FR(MatriceSparsa matriceH, int[] pagPendenti, MatriceSparsa vettoreP){
        H = matriceH; a = pagPendenti; pi_ZERO = vettoreP;
}

public static void main(String[] args) {

 ArsConn ac = new ArsConn();

 if (ac.getCheck()) {
  FR fr = new FR(ac.getMatrix(), ac.getHanger(), ac.getVector());
  System.out.println();
  int numIterazioni = 0; // si vuol conoscere il numero di iterazioni

  if(MatriceSparsa.normalizzato1(fr.pi_ZERO)
   && MatriceSparsa.sottostocastica(ac.getMatrix())) {
       System.out.print(" ArsVector E' normalizzato. \n");
       System.out.println(" ArsMatrix E' sottostocastica.\n");

       MatriceSparsa alfaH = MatriceSparsa.prodottoScalare(fr.ALFA, fr.H);
       MatriceSparsa addendo3 =
               MatriceSparsa.prodottoScalare(fr.CO_ALFA, fr.pi_ZERO);

       fr.pi_NEXT = fr.pi_ZERO;
       do {
```

```
            fr.pi_PREV = fr.pi_NEXT;
            MatriceSparsa addendo1 =
               MatriceSparsa.prodottoVeloce_vRmS(fr.pi_PREV, alfaH);
               MatriceSparsa addendo2 =
                  MatriceSparsa.prodottoScalare(fr.ALFA,
                  MatriceSparsa.prodotto_vRa(fr.pi_PREV, fr.a));
              fr.pi_NEXT =
                   MatriceSparsa.somma2vettoriR(addendo1, addendo2);
              fr.pi_NEXT =
                   MatriceSparsa.somma2vettoriR(fr.pi_NEXT, addendo3);


                  // Comunica all'utente le iterazioni fatte, di 10 in 10
                  if(numIterazioni%10== 0) {
                   System.out.print(numIterazioni + " - ");
                   if (MatriceSparsa.normalizzato(fr.pi_NEXT))
                          System.out.print(" pi_NEXT normalizzato ");
                   else {
                          System.out.print(
                             " pi_NEXT NON normalizzato: break! ");
                           break;
                   }
                  System.out.println();
          }
          numIterazioni++;
}
while(!MatriceSparsa.converge(fr.pi_PREV, fr.pi_NEXT, fr.EPSILON)
   && numIterazioni < 201);
          System.out.println();
          System.out.print(" PI_NEXT = ");
          if (MatriceSparsa.normalizzato(fr.pi_NEXT)) {
                  System.out.println("\nconverge");
```

```
                    ArsConn.updateSup(fr.pi_NEXT); // scrittura dei risultati nel db
                }
                else { System.out.println("\ndiverge"); }
        }
        else {
                if (!MatriceSparsa.normalizzato1(fr.pi_ZERO))
                    System.out.println(" ArsVector NON e' normalizzato.\n");
                if (!MatriceSparsa.sottostocastica(ac.getMatrix()))
                    System.out.println(" ArsMatrix NON e' sottostocastica.\n");
        }
        System.out.println();
        System.out.println("\n" + numIterazioni + " iterazioni\n");
        System.out.println("DONE.\n");

        }
    }
}
```

9.4 Conclusioni

Durante la scrittura dell'algoritmo (e in generale durante la stesura della tesi) si è vista una panoramica di quello che può essere la gestione dei progetti informatici: si è apprezzato come la pianificazione dei progetti sia sicuramente un'attività molto complessa, da realizzare mediante un continuo controllo sullo stato di avanzamento delle diverse attività.

Gli obiettivi sono stati chiari, ben definiti e condivisi sin dall'inizio e si è avuto modo di vedere come gestire quei fattori ostativi al raggiungimento dei risultati di progetto.

Altri progetti di cui si potrebbe prendere considerazione sono quelli relativi a produrre altre classi per poter estendere le funzionalità presenti o per provvedere a nuove funzionalità, come il suggerimento di tag, la risoluzione di ambiguità e sinonimie

nell’assegnazione di tag, la rilevazione di comunità all’interno del sito, filtri per l’analisi di spamming (sempre relativamente ai tag) e analisi del ranking attribuito dall’algoritmo mediante contributo degli utenti (tramite confronto di valori attribuiti dall’algoritmo contrapposti a valori di importanza indicati dagli utenti).

Appendice A
Richiami sulle matrici e metodo delle potenze

Il metodo delle potenze è quello che è stato utilizzato per il calcolo del FolkRank per ArsMeteo. Per comprenderlo sono necessarie poche nozioni sulle matrici.

Matrici

Le matrici si possono definire come un insieme di elementi che hanno la particolarità di essere allineati su un certo numero di righe e contemporaneamente su un certo numero di colonne. Ad esempio la matrice

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1c} \\ a_{21} & a_{22} & \dots & a_{2c} \\ \dots & \dots & \dots & \dots \\ a_{r1} & a_{r2} & \dots & a_{rc} \end{pmatrix}$$

ha i suoi elementi allineati su un numero di righe pari a r e su un numero di colonne pari a c.

Ordine delle matrici

Nella precedente matrice, il numero di righe (r) e il numero di colonne (c) si dicono dimensioni di A o ordine di A e si scrive $A \in R^{r,c}$ per denotare il fatto che A ha r righe e c colonne i cui elementi sono valori reali.

Se r = c, A si dice quadrata e si può definire il concetto di potenza di matrice con esponente intero.

Vettori

Quando una matrice è formata da una sola riga e c > 1 colonne (o viceversa) la si chiama “vettore” (vettore riga o vettore colonna).

Moltiplicazione

Fra le varie operazioni definibili sulle matrici, per il seguito ha importanza considerare soltanto la moltiplicazione fra matrici.

Consideriamo l’insieme dei numeri reali. Il prodotto è definito fra due matrici A e B soltanto nel caso in cui le loro dimensioni abbiano una particolare proprietà, cioè soltanto nel caso in cui sia $A \in R^{r,p} \quad e \quad B \in R^{p,c}$; il risultato della moltiplicazione sarà la matrice $AB = C \in R^{r,c}$. Se le dimensioni dei fattori non hanno le caratteristiche volute la moltiplicazione non si può fare. La conseguenza è che in genere la proprietà commutativa non vale, cioè in generale $AB \neq BA$. Per questo si utilizzano le espressioni “moltiplicazione a sinistra” e “moltiplicazione a destra”.

Se le matrici A e B sono entrambe quadrate e dello stesso ordine, allora il prodotto è sempre definito. In questo caso, una matrice particolare è la matrice identità, indicata da I, che è la matrice in cui gli elementi che sono sulla diagonale (quelli per cui i valori di r e c coincidono) hanno tutti il valore 1.

Un altro tipo di moltiplicazione che comparirà in qualche formula nel seguito è la moltiplicazione fra uno scalare e una matrice, denotata da lA, dove il primo termine è uno scalare e il secondo è una matrice: il risultato è la matrice A i cui elementi sono tutti moltiplicati per lo scalare dato.

Determinante

Sul determinante di una matrice A è sufficiente dire che esso è denotato da det(A) ed è uno scalare. Il numero di operazioni necessarie per il suo calcolo possono essere proibitive per matrici di grandi dimensioni.

Dipendenza e indipendenza lineare

Dati i vettori $X, V_1, ..., V_n$, tutti dello stesso tipo (cioè tutti vettori riga o tutti vettori colonna) e tutti dello stesso ordine, se esistono delle costanti reali $d_1, ..., d_n$ tali che $X = d_1V_1 + ... + d_nV_n$ allora X si dice combinazione lineare dei vettori V_i.

I vettori V_i possono formare un insieme di vettori:

- linearmente dipendenti se esiste almeno un d_i non nullo e X è il vettore nullo

- linearmente indipendenti se tutti i d_i sono nulli e anche X è il vettore nullo.

Autovalori, autovettori ed equazione e polinomio caratteristici

Un vettore colonna X si dice autovettore di una matrice quadrata A se esiste uno scalare $l \,/\, AX = lX$. In questo caso lo scalare si dice autovalore di A. Un autovalore può essere l'autovalore nullo, ma un autovetture non può essere l'autovettore nullo.

L'equazione secolare (o caratteristica) di una matrice quadrata A è il polinomio di grado n-esimo $\det(A - lI) = 0$, dove il valore tra parentesi si chiama polinomio caratteristico di A.

Per calcolare il determinante sopra riportato occorrono O(n!) moltiplicazioni; anche il calcolo delle radici del polinomio richiede un numero di operazioni eccessivo: i problemi sono intrattabili. Allora per poter calcolare gli autovalori di grandi matrici si utilizzano algoritmi numerici (si tratta di metodi iterativi comprendenti una condizione di terminazione se i risultati convergono e un limite superiore al numero di iterazioni permesse nel caso ci sia convergenza) che permettono di ottenere autovalori e loro autovettori. Il più semplice di questi metodi è appunto il metodo della potenza.

Una proprietà importante è che autovettori corrispondenti ad autovalori diversi sono linearmente indipendenti e quindi, quando vengono uniti tutti gli insiemi

corrispondenti a tutti gli autovalori si ottiene un insieme massimale di autovettori della matrice data che sono linearmente indipendenti.

Molteplicità

La molteplicità di una radice a di un polinomio p(x) è il numero di volte che p(x) può essere diviso per (x-a).

La molteplicità di un autovalore a, indicata da mult(a), è la molteplicità di a in quanto radice del polinomio caratteristico.

Catene

Sono necessarie poche definizioni preliminari.

FORMA ORDINATA PER RIGHE

Riga zero in una matrice è quella riga della matrice che ha elementi tutti nulli.

Riga non-zero è quella che contiene almeno un elemento non nullo.

Matrice zero è quella matrice i cui elementi sono tutti nulli.

Una matrice si dice in forma ordinata per righe se soddisfa a quattro condizioni:

1. Tutte le righe non zero sono sopra alle righe zero
2. In una riga non-zero l'elemento non-zero più a sinistra è l'unità
3. Se il primo elemento non-zero di una riga è in colonna c, allora nelle righe che seguono tutti gli elementi della colonna c sono nulli
4. Il primo elemento non nullo di una riga non-zero si trova in una colonna più a destra rispetto a quella del primo elemento non nullo di una riga precedente

Le matrici che soddisfano le precedenti condizioni si dicono ordinate per riga.

RANGO

Il rango di una matrice è il numero di righe non-zero che contiene dopo essere stata portata in forma ordinata per righe.

AUTOVETTORE DI RANGO M

Un vettore X_m dice autovetture (destro) di rango m per una matrice A e per l'autovalore associato l se $(A - lI)^m X_m = \bar{0} \quad ma \quad (A - lI)^{m-1} X_m \neq \bar{0}$, dove la notazione $\bar{0}$ indica un vettore nullo.

CATENA

E' l'insieme $\{X_m, X_{m-1}, ..., X_1\}$ linearmente indipendente generato dal vettore di rango m X_m, dove $X_j = (A - lI)X_{j+1}, (j = m-1, m-1, ..., 1)$

LUNGHEZZA DELLA CATENA

E' la cardinalità dell'insieme precedente.

Matrici stocastiche ed ergodiche e catene (finite) di Markov

Una matrice si dice stocastica se tutte le sue somme di riga (o di colonna) sono uguali all'unità.

Una matrice ergodica è una matrice stocastica, irriducibile e aperiodica. In una matrice ergodica, l'autovalore $l = 1$ è l'unico autovalore di modulo 1 e se tale valore ha molteplicità k allora esistono k autovettori linearmente indipendenti ad esso corrispondenti.

Una catena di Markov a N stati è formata da due insiemi: un insieme di oggetti e un insieme finito di N stati diversi, dove N è fissato a priori e tale che:

1. a un certo tempo un oggetto si trova in uno degli N stati (che possono essere differenti per ogni oggetto)

2. la probabilità che un oggetto passi da uno stato all'altro in un intervallo di tempo dipende solo dagli stati di inizio e fine.

La matrice $P \in R^{NxN}$ che indica la probabilità considerata nel punto due è stocastica e l'elemento (i,j) della potenza m-esima di P rappresenta la probabilità che un oggetto passi dallo stato i allo stato j in m intervalli di tempo. Per m tendende all'infinito la distribuzione di una catena di Markov è uguale all'unico autovettore stocastico associato all'autovalore, indipendentemente da ogni distribuzione iniziale.

Matrici irriducibili

Anche qui alcune definizioni preliminari.

MATRICI NON NEGATIVE

Sono matrici i cui valori sono tutti reali maggiori od uguali a 0.

MATRICE PERMUTAZIONE

Sono quelle che si ottengono dalla matrice identità ridisponendone le righe.

MATRICE TRASPOSTA

La matrice ottenuta scambiando righe con colonne, indicata con una lettera T come esponente.

MATRICI RIDUCIBILI E IRRIDUCIBILI

Una matrice non negativa A è riducibile se esiste una matrice permutazione P tale che $PAP^T = \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{pmatrix}$ in cui sia A_{11} *che* A_{22} hanno ordine minore di A; se non esistono tali matrici permutazione allora A si dice irriducibile (una matrice irriducibile non può avere righe o colonne nulle).

Matrici periodiche e aperiodiche

Dato un insieme finito di stati S = { 1, 2, … , m } e una matrice in uno stato i, si dice periodo della matrice il massimo comune divisore degli n per i quali $p_{i,i}^{(n)} \neq 0;$ cioè $p = MCD\{n \in N \,/\, p_{i,i}^{(n)} > 0\}$.

Per p = 1, la matrice viene detta aperiodica

Il metodo delle potenze

Consiste nello scegliere un vettore X e formare la serie $X, AX, A^2X, A^3X, \ldots$ che converge ad un autovettore di A dal quale si ricava anche l'autovalore dominante (si dice dominante perché è quello con maggior valore assoluto)

Traccia della dimostrazione

Sia data A, quadrata, di ordine n, a valori reali, con n autovalori reali tali che $|l_1| > |l_2| \geq |l_3| \geq \ldots \geq |l_n|$ e gli autovettori V_i corrispondenti formino un insieme linearmente indipendente.

Con queste premesse, per ogni vettore X di dimensione n esisteranno delle costanti d_i non tutte nulle tali che $X = d_1V_1 + \ldots + d_nV_n$, da cui, moltiplicando più volte a sinistra per A si ottiene $A^kX = d_1l_1^kV_1 + \ldots + d_nl_n^kV_n$, da cui, dividendo per l'autovalore dominante $\frac{1}{l_1^k}A^kX = d_1V_1 + \ldots + d_n\left(\frac{l_n}{l_1}\right)^kV_n$

Bibliografia

Libri

Ferry Bite, Claudio Parrini, “I motori di ricerca nel caos della rete - kit di sopravvivenza”, ShaKe, Milano, 2001.

Google's “PageRank and Beyond: The Science of Search Engine Rankings”. Princeton University Press, 2006.

Roberto Marangoni, Alessandro Cucca, “Motori di ricerca”, Hoepli, Milano 1999.

Tim Berners-Lee, “L’architettura del nuovo Web - Dall’inventore della rete il progetto di una comunicazione democratica”, Feltrinelli, Milano, 2001.

Morrison P. Jason, “Tagging and Searching: Search Retrieval Effectiveness of Folksonomies on the Web”, College of Communication and Information of Kent State University, Master Thesis, 2007.

Articoli

BAO et al, 2007,
Optimizing Web Search Using Social Annotations,
http://www2007.org/papers/paper397.pdf.

BERNERS-LEE et al, 2006
The Semantic Web Revisited
http://eprints.ecs.soton.ac.uk/12614/01/Semantic_Web_Revisited.pdf

CATTUTO et al, 2007,
Network Properties of Folksonomies,
http://www2007.org/workshops/paper_13.pdf.

DIEDERICH et al, 2006,
Finding Communities of Practice from User Profiles Based On Folksonomies,
http://www.l3s.de/~diederich/Papers/TBProfile-telcops.pdf.

HOTHO et al, 2006,
FolkRank: A Ranking Algorithm for Folksonomies,
http://www.kde.cs.uni- http://kassel.de/stumme/papers/2006/hotho2006folkrank.pdf.

HOTHO et al, 2006,
Mining Association Rules in Folksonomies,
http://www.kde.cs.uni- kassel.de/hotho/pub/2006/schmitz2006asso_ifcs.pdf.

HOTHO et al, 2007,
Tag Recommendations in Folksonomies,
http://www.kde.cs.uni-
kassel.de/stumme/papers/2007/jaeschke07tagrecommendationsKDML.pdf.

HOTHO et al, 2007,
TRIAS - An Algorithm for Mining Iceberg Tri-Lattices,
http://portal.acm.org/citation.cfm?id=1193256.

HWANG S. Hyung, 2007,
A Triadic Approach of Hierarchical Classes Analysis on Folksonomy Mining,
http://paper.ijcsns.org/07_book/200708/20070827.pdf.

SCHMITZ C., 2006,
Small World Folksonomies Clustering in Tri-Partite Hypergraphs,
https://kobra.bibliothek.uni-kassel.de/bitstream/urn:nbn:de:hebis:34-
2006120415997/1/techreport.pdf.

STOCK G. Wolfgang, 2007,
Folksonomies and science communication,
http://wwwalt.phil-fak.uni-
duesseldorf.de/infowiss/admin/public_dateien/files/1/1194272247inf_servic.pdf.

Siti web (sitografia)

Eioba.
http://www.eioba.com/a69792/the_google_pagerank_algorithm_in_126_lines_of_python.

Google Page Rank.
http://www.ianrogers.net/google-page-rank/.

IBM.
http://www-128.ibm.com/developerworks/library/x-plansemantic/.

IBM.
http://www.ibm.com/developerworks/edu/x-dw-x-semanticweb.html.

MITRE.
http://www.mitre.org/news/the_edge/summer_04/manola.html.

MOTORICERCA.
http://www.motoricerca.info/articoli/google-dance.phtml

MUSINGS.
http://magia3e.wordpress.com/2008/03/08/ontologies-the-world-beyond-web-20/

Stalkk.ed.
http://www.stalkked.com/2008/04/07/web-tagging-tagging-semantico-folksonomie-folkstologie/

Wikipedia.
http://it.wikipedia.org/wiki/Web_semantico.

Wikipedia.
http://it.wikipedia.org/wiki/PageRank

Wikipedia.
http://it.wikipedia.org/wiki/Tag_(metadato).