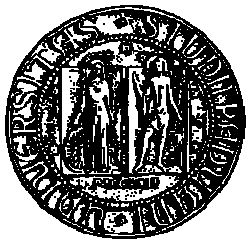


UNIVERSITÀ DEGLI STUDI DI PADOVA
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria Informatica

TESI DI LAUREA SPECIALISTICA

ESTENSIONE MOZILLA PER L'OPAC DELL'UNIVERSITÀ DI PADOVA CON ESPANSIONE DELLA QUERY IN GOOGLE

RELATORE: Prof. Massimo Melucci

LAUREANDO: Alex Villatora

A.A. 2009-2010

Indice

Introduzione

Questo elaborato presenta un lavoro che si suddivide principalmente in due parti: nella prima parte c'è stato uno studio abbastanza approfondito dell'online advertising e delle sue problematiche; mentre nella seconda parte è stata realizzata un estensione per Mozilla Firefox che effettua una ricerca nel Catalogo dell'OPAC (Online Public Access Catalogue) del sistema bibliotecario padovano e che espande la query iniziale grazie ad alcuni giudizi di rilevanza espressi dall'utente. L'idea è quella di realizzare un'estensione che sfrutti i principi dell'online advertising (in particolare della sponsored search) per trovare risultati nell'OPAC, che possano essere utili per soddisfare l'esigenza informativa espressa dall'utente in Google. Lo scopo dell'estensione è, in primo luogo, quello di permettere all'utenza di conoscere strumenti, come, appunto, i cataloghi OPAC, normalmente non utilizzati, ma di grande qualità per il materiale a disposizione. La seconda parte dell'estensione, poi, permette di migliorare la ricerca dell'utente senza che lui pensi direttamente alla query da utilizzare ma semplicemente sfruttando alcune sue indicazioni. L'utente spesso non riesce ad esprimere, subito, in maniera chiara le sue esigenze, per cui un processo di espansione della query può aiutarlo nel cercare documenti utili ai suoi scopi. Un'altro motivo che ha portato allo sviluppo della seconda parte dell'estensione è il problema del "topic drift" cioè quando risultati relativi ad argomenti diversi fanno riferimento alla stessa query. L'estensione modifica

leggermente l’interfaccia del motore di ricerca e provvede a migliorare i risultati della sua ricerca.

I cataloghi OPAC sono strumenti disponibili attraverso siti specifici che però non sono molto conosciuti; per questo l’estensione permette di sfruttare lo strumento più comunemente usato dagli utenti, il motore di ricerca, per consigliare i materiali disponibili nell’OPAC.

La relazione si sviluppa in sei capitoli:

- **1. Online Advertising** : descrive in modo abbastanza dettagliato il mondo della pubblicità sul web, in particolare spiega come sono gestiti gli annunci pubblicitari che si vedono a fianco dei risultati nei motori di ricerca e quali sono gli attori in gioco.
- **2**. **Click Fraud** : questo capitolo descrive il problema più grosso che affligge il mondo della pubblicità sul web indicando alcune contromisure spesso utilizzate allo stato attuale.
- **3**. **Catalogo: l’OPAC dell’ateneo** : introduce il catalogo OPAC utilizzato per recuperare i dati poi visualizzati dall’estensione con una descrizione dettagliata degli strumenti offerti dal suo sito.
- **4**. **Un problema e la sua soluzione: il Topic drift** : è un capitolo prettamente teorico che espone gli strumenti a disposizione per risolvere il problema del topic drift, problema che l’estensione si pone di superare con la query expansion.
- **5. L’applicazione** : descrive in maniera dettagliata le fasi dell’estensione, i file che la compongono e i singoli algoritmi implementati anche grazie all’uso del codice sorgente.
- **6 Conclusioni :** conclude l’elaborato riassumendo il lavoro svolto, sia teorico che pratico e fornisce delle indicazioni per quanto riguarda possibili soluzioni alternative sulla scelta del motore di ricerca o dell’OPAC di riferimento.

1 Online Advertising

La pubblicità (in inglese *advertising*) è una forma di comunicazione che ha tre principali obiettivi:

- far giungere messaggi commerciali e attirare così nuovi clienti;
- informare i potenziali clienti di prodotti e/o servizi e su come ottenerli e come usarli;
- fare incrementare i consumi di certi prodotti e servizi per consolidare la posizione del marchio dei produttori nel mercato.

Per trasmettere i messaggi pubblicitari vengono utilizzati tutti i principali mezzi di comunicazione tra i quali la televisione, la radio, le riviste, i giornali, Internet ed i cartelloni pubblicitari.

La pubblicità su Internet sta assumendo un ruolo sempre più importante nel mondo dell'advertising. Per capire quanto è importante basta pensare che nel 2008 la spesa per la pubblicità in rete è stata di 65 miliardi di dollari, circa il 10% dell'intera spesa pubblicitaria di tutto il mondo su tutti i media e che queste cifre sono previste in aumento per i prossimi anni [IDC 2008].

La pubblicità su Internet (*online advertising*) è una forma di promozione che sfrutta il Web per trasmettere messaggi in grado di attrarre utenti interessati ad eventuali

acquisti. Esistono diversi tipi di pubblicità in rete che variano per dimensione e formato; si va dal semplice link testuale con annuncio annesso, ai banner, ai pop-up.

a)

b)

c)

Fig 1.1: Esempi di tipi e formati pubblicitari: a) messaggi testuali+link, b) banner e c) pop-up

Scopo dell'online advertising è quello di generare dei profitti per chi visualizza delle pubblicità nelle pagine del proprio sito, e creare ROI (*Return on Investment*) per gli inserzionisti. Il ROI indica la redditività e l'efficienza economica di un investimento [Wik], in questo caso il ROI permette di valutare se la spesa (investment) fatta per esporre delle pubblicità porta a dei ricavi (return), sperando che quest'ultimi siano maggiori dei costi di gestione della stessa.

Le parti coinvolte, nel processo pubblicitario, sono due:

1. l'inserzionista (*advertiser*), cioè chi decide di investire nella pubblicità,
2. e chi mette a disposizione, dietro retribuzione, dello spazio per le pubblicità nelle proprie pagine Web (*publisher*).

Publisher ed advertiser possono comunicare direttamente tra loro per accordarsi sui vari aspetti che riguardano lo sviluppo di una campagna pubblicitaria, comunque nella maggior parte dei casi, esiste un intermediario tra le parti, l'*advertising network* (*ad network*). L'ad network ha il ruolo di mediare tra coloro che vogliono ospitare della pubblicità nel proprio sito e coloro che vogliono fare pubblicità dei propri prodotti e/o servizi. In questo modo non esiste un vincolo unico tra chi pubblica e cosa viene pubblicato; cioè non è detto che la stessa pagina pubblichi sempre lo stesso messaggio, in quanto il publisher dà la sua disponibilità ed è l'ad network che sceglie il messaggio pubblicitario da inserire nella pagina tra quelli ricevuti come proposta dai vari advertiser e accettati dal publisher come possibili candidati (la stessa pagina può presentare messaggi diversi se caricata in momenti differenti).

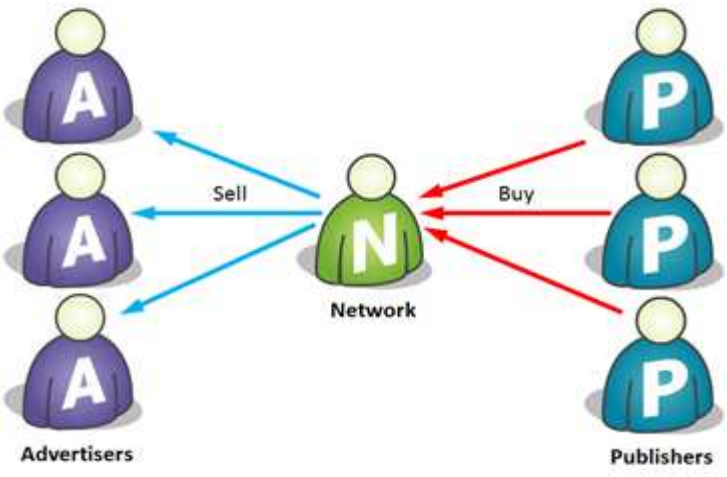


Fig 1.2: I ruoli in gioco, in particolare quello dell'ad network [Will].

Le più famose ad network gestiscono anche dei motori di ricerca e sono Google e Yahoo; queste due network coprono circa l'80% del mercato mondiale. Google e Yahoo forniscono diversi servizi ai publisher e agli advertiser; raccolgono i nomi dei publisher che desiderano esporre della pubblicità nei propri siti e gestiscono i link e i messaggi da mostrare nelle loro pagine; inoltre, in base ai vincoli economici stabiliti tra le parti, danno una parte degli incassi ricevuti dagli advertiser, ai publisher, che nel loro sito hanno esposto dei messaggi pubblicitari di tali advertiser. Per quanto riguarda gli advertiser, le ad network, raccolgono le varie campagne pubblicitarie che questi vogliono sviluppare e usano gli annunci pubblicitari, che le compongono, per visualizzarli nelle pagine dei risultati dei motori di ricerca o nei siti dei publisher. Quando i messaggi pubblicitari vengono presentati nelle pagine dei publisher, passando attraverso un ad network, sono riconoscibili dal fatto che vicino, a tali annunci, si trova una scritta in piccolo che segnala l'ad network di provenienza. I messaggi pubblicitari che compaiono nelle pagine dei publisher non sono collegati ad una ricerca (come nei motori di ricerca), ma sono legati al contesto espresso dal contenuto della pagina stessa in cui sono visualizzati; si tratta di pubblicità contestuale (*contextual advertising)*.Il contextual advertising è una forma di pubblicità che compare nei siti dei publisher, dove i messaggi pubblicitari da visualizzare vengono scelti in maniera automatica dall'ad network in base al contenuto del sito stesso [Wik].

1.1 Sponsored Search

Le ad network, come già accennato, sono nella maggior parte dei casi, delle compagnie che gestiscono un motore di ricerca e lo sfruttano per pubblicare della pubblicità nelle pagine dei risultati, le cosiddette ricerche sponsorizzate (*sponsored search*).

Sponsored search è una forma di pubblicità legata ai motori di ricerca che prevede che gli advertiser paghino le ad network per il traffico che arriva ai loro siti dai motori di ricerca [Sci]. Quando un utente ha l'esigenza di trovare qualcosa nella rete Internet, sfrutta spesso i motori di ricerca. Nei motori di ricerca, quando si inserisce una query, viene presentata una lista di link utili uno sotto l'altro. Nella pagina dei

risultati viene spesso presentato sulla destra un insieme di link evidenziati come "sponsored links". I link che vengono scelti, per questo insieme, dovrebbero essere inerenti l'argomento della ricerca o in qualche modo collegati ad essa; non sempre questo, però, si verifica a causa del procedimento usato per selezionare i link e per il comportamento non sempre regolare degli advertiser. Tutto ciò rende le procudure di selezione dei "sponsored links" un tema caldo nel campo della ricerca per quanto riguarda l'online advertising.

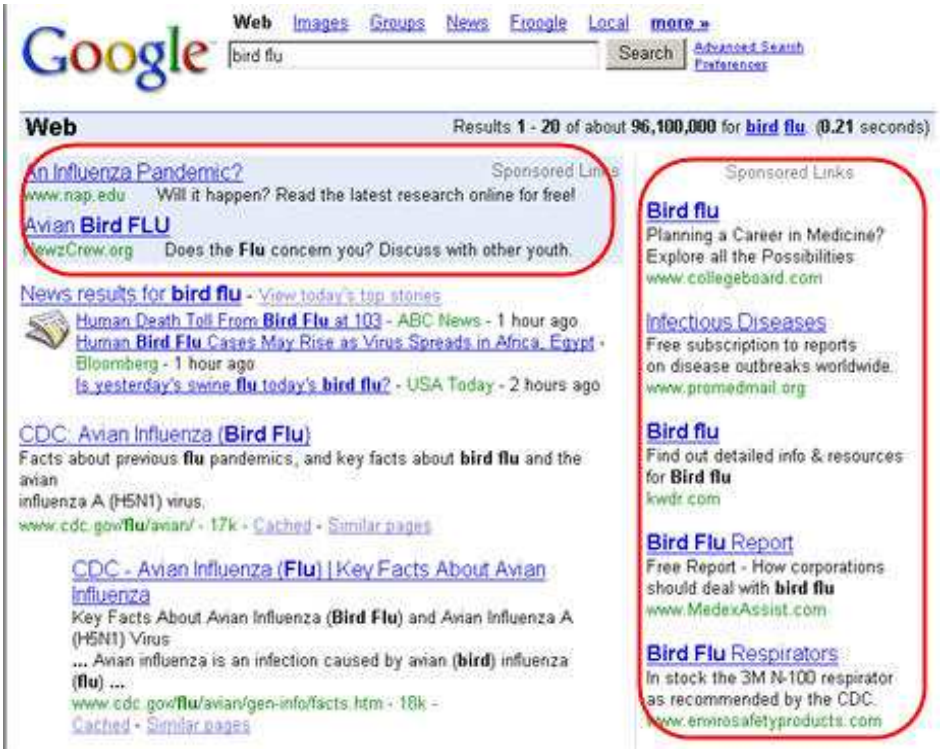


Fig 1.3: Esempio di sponsored search con sponsored links

Se l'utente clicca su uno di questi link "sponsorizzati" viene mandato al sito dell'advertiser intestatario del link, il quale, poi, per questo, pagherà l'ad network, titolare del motore di ricerca.

Per quanto riguarda la disposizione dei link "sponsorizzati" all'interno della pagina dei risultati bisogna dire che gli annunci non vengono visualizzati in maniera casuale; i messaggi compaiono secondo un ordine ben stabilito da una classifica che viene stilata valutando quanto, ciascun advertiser, è disposto a pagare per far visualizzare la sua pubblicità in quel contesto e una stima dei click che può ricevere.

Il modello utilizzato per classificare i messaggi comporta un variare della spesa per l'inserzionista. I modelli sviluppati per decidere il *ranking* degli annunci sono principalmente due:

- *Generalized First Prize Auction* (1997, GoTo Overture poi diventata Yahoo Search)*;*
- *Generalized Second Price Auction* (2002 Google AdWords).

1.1.1 Generalized First Price Auction

Il primo modello per redigere la classifica degli annunci si sviluppa su un concetto simile ad un'asta. Gli advertiser decidono quali parole chiave possono essere legate agli argomenti trattati nei loro annunci e quindi di riflesso ai prodotti e/o servizi da loro forniti; una volta scelte queste *keyword,* decidono e comunicano all'ad network la somma che vogliono puntare su ciascuna parola. Quando l'utente invia una query al motore di ricerca, l'ad network controlla le parole inserite dall'utente, e trova una corrispondenza tra quest'ultime e le keyword scelte dagli advertiser. Per ogni keyword selezionata recupera gli annunci ad essa legati e confronta le somme che ogni advertiser ha deciso di "puntare". La classifica viene così decisa mettendo al primo posto il messaggio, con link, che è legato alla somma più alta, a seguire quello con la somma subito successiva e così via in maniera decrescente per cifra "puntata". Conquista la prima posizione chi è disposto a pagare di più; infatti le cifre "puntate" indicano la quantità di denaro che un advertiser pagherà all'ad network nel caso in cui l'utente clicchi sul suo link. [IASel].

1.1.2 Generalized Second Price Auction

Il modello in assoluto più popolare al momento nella sponsored search è quello che si chiama Generalized Second Price Auction [SSA]; anche Google e Yahoo usano questo modello, per la verità con una piccola modifica, almeno per quel che riguarda Google [Yah] [Gog]. Il modello si basa sempre sul concetto di asta, come il precedente. Tutto il modello segue il procedimento descritto prima con l'associazione, da parte degli advertiser, delle keyword agli annunci pubblicitari, l'estrazione delle parole chiave, da parte dell'ad network, dalla query di ricerca

dell'utente, le "puntate", degli advertiser, sulle diverse keyword. Anche in questo caso il primo posto viene assegnato all'advertiser che ha deciso di "puntare" la cifra più alta; la differenza sostanziale sta nella cifra che un advertiser deve pagare all'ad network in caso di *click* sul suo link. Quando avviene un click l'advertiser titolare di quel link deve pagare una somma pari alla "puntata" dell'advertiser che occupa la posizione in classifica immediatamente inferiore più una piccola aggiunta [IASel]. Il primo in classifica paga la "puntata" del secondo più un delta, il secondo paga la "puntata" del terzo e così via a scalare per il resto della classifica. Come detto questo modello è sfruttato anche da Google, che però, per quanto riguarda la classifica, tiene conto anche di un rapporto detto CTR (*Click-Through-Rate*) che serve per misurare il successo di una pubblicità. Il CTR è dato dal rapporto tra il numero di click ricevuti dall'annuncio nella pagina ed il numero di volte che il messaggio è stato visualizzato nella pagina stessa, il tutto moltiplicato per cento per esprimerlo in percentuale.

CTR = (Clicks/Impression) x 100

Più alto è il CTR e maggiore peso avrà in classifica l'annuncio. Spesso nelle campagne pubblicitarie di questo tipo, gli advertiser, fissano dei limiti per quanto riguarda la cifra massima da "puntare" per una keyword e il budget settimanale e/o mensile da spendere nella campagna stessa.

1.2 Modelli di pagamento

Per la sponsored search, come pure per il contextual advertising, esistono diversi modelli di pagamento che sono stati sviluppati negli anni. Vediamo questi modelli seguendo una linea cronologica:

1.2.1 CPM – Cost Per Thousand (1994)

Questo primo modello prevede che l'advertiser paghi, una quota stabilita a priori, ogni volta che il suo messaggio pubblicitario viene visualizzato in un sito (M nell'acronimo sta per il numero mille scritto in romano) [POAd]. È anche detto CPI

(Cost Per Impression), dove per impressione si intende una singola apparizione del messaggio in una pagina web. Naturalmente le operazioni come il ricaricamento di una pagina o la visualizzazione della pagina dovuta alla freccia “Indietro” del browser non vengono conteggiate nelle impressioni. Questo è un modello usato principalmente con i banner e che va a vantaggio più dei publisher che possono sfruttare particolari situazioni in cui si presenta una grande quantità di traffico web e quindi molte impressioni. Questo modello, come tutti, presenta delle possibili situazioni di pericolo quali *l’impression spam.* Lo spam nasce da richieste http che riguardano pagine contenenti pubblicità, ma che non necessariamente derivano da un’operazione intenzionale di un utente; esistono programmi, i *web page crawler*, che fanno moltissime richieste http di pagine contenenti messaggi pubblicitari e tutto questo traffico non è utile al fine del conteggio delle impressioni in quanto queste pagine non vengono richieste intenzionalmente ma in maniera automatica, senza un vero interesse verso la pagina stessa da parte dell’utente, per cui è molto importante distinguere questo traffico da quello prodotto da veri utenti. Un altro punto importante sta nel fatto che un advertiser non riceve nessun feedback quando un’impressione del suo messaggio avviene in una pagina web, perciò deve affidarsi completamente ai dati che gli arrivano dall’ad network. Nel 2000, Google lanciò AdWords, una piattaforma per l’advertising, basata sul modello CPM, in cui i messaggi testuali erano selezionati secondo le parole inserite dall’utente per compiere una ricerca. Google inizialmente fece dei contratti con gli advertiser per mostrare le loro pubblicità in cima alla lista dei risultati delle ricerche basandosi sul modello CPM, più tardi cominciò a mettere all’asta i posti sul lato destro della pagina dei risultati ed eventualmente anche i tre in cima alla lista dei risultati. In Google queste tre posizioni vengono tutt’ora assegnate esclusivamente agli annunci pubblicitari che superano un certo punteggio di qualità e una determinata offerta minima [GoAd]. Una nuova forma di mercato in cui far pagare gli advertiser non in base al numero di impressioni, ma in base al numero di volte che un utente clicca sul messaggio pubblicitario ha cominciato a svilupparsi negli anni successivi.

Fig 1.4: La pagina di accesso ad AdWords di Google

1.2.2 CPC – Cost Per Click (1997)

Questo modello è quello che ha avuto più successo nel mondo dell'advertising; prevede che l'advertiser paghi, una cifra pattuita, ogniqualvolta un utente esegue un click su una sua pubblicità e venga così reindirizzato nel sito dell'advertiser stesso [POAd]. La quota che l'advertiser deve pagare all'ad network, e al publisher di conseguenza, è stabilita dal meccanismo di asta che viene usato per assegnare il ranking ai messaggi. Nel caso in cui il messaggio sia pubblicato su un sito, senza il coinvolgimento di un ad network, la quota da pagare, da parte dell'advertiser, è fissa e stabilita direttamente col publisher mediante contratto diretto. Il modello può essere anch'esso vittima di abusi tramite il *click fraud*.

Da questo modello può trarre vantaggio il publisher "gonfiando" il numero di click ricevuti sui link delle proprie pagine web (si veda capitolo 2); e l'advertiser quando è in concorrenza per un posto in classifica nella sponsored search (capitolo 2). Questo modello deve considerare in qualche modo anche i click non validi, molto spesso

dovuti, per esempio, ad un doppio click su un link, ed eliminarli dal conteggio valido per il pagamento.

Nel 2002, Google ha rilanciato AdWords come una piattaforma CPC; Google comunque non solo considera le "puntate" degli advertiser, ma considera anche il CTR degli stessi. Una posizione nella pagina viene così allocata all'advertiser che promette di avere i migliori risultati economici secondo i calcoli dell'ad network [GoAd]. In questo modo il piazzamento dei messaggi pubblicitari non è solo il risultato di grosse spese, ma è una conseguenza della qualità del messaggio stesso; ogni click fatto da un utente serve come voto implicito della rilevanza della pubblicità nei confronti dell'esigenza espressa dall'utente. Grazie a questa soluzione l'utente può ricevere messaggi più allettanti e Google può incrementare i suoi ricavi migliorando la scelta delle pubblicità.

Sempre Google nel 2003 ha lanciato AdSense, un prodotto per la pubblicità in Internet che permette ai publisher di monetizzare lo spazio libero delle loro pagine web permettendo a Google di inserire delle pubblicità contestuali in questi spazi [GoSe]. Per fare in modo che i messaggi siano rilevanti per l'utente, le pagine dei publisher vengono analizzate da AdSense per determinare quali sono gli argomenti trattati e per scegliere poi degli annunci pubblicitari che riguardino tali argomenti in modo da poterli inserire nelle pagine del publisher stesso. Quando una delle pubblicità nelle pagine del publisher viene cliccata da un utente, Google gira al publisher una parte della somma ricevuta dall'advertiser.

Mentre, da una parte la creazione di AdSense ha aiutato i publisher a guadagnare qualcosa grazie allo sviluppo del contenuto delle loro pagine web tramite le pubblicità, dall'altra ha introdotto un incentivo in più per tentare nuove attività di click fraudolento (capitolo 2).

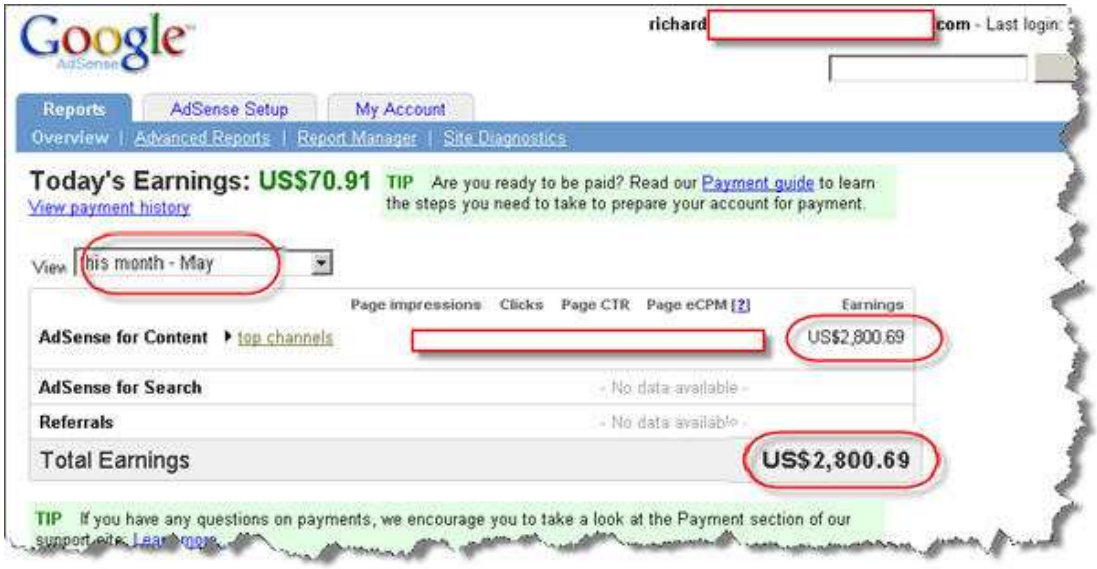


Fig 1.5 a): Una pagina di AdSense di Google per i publisher

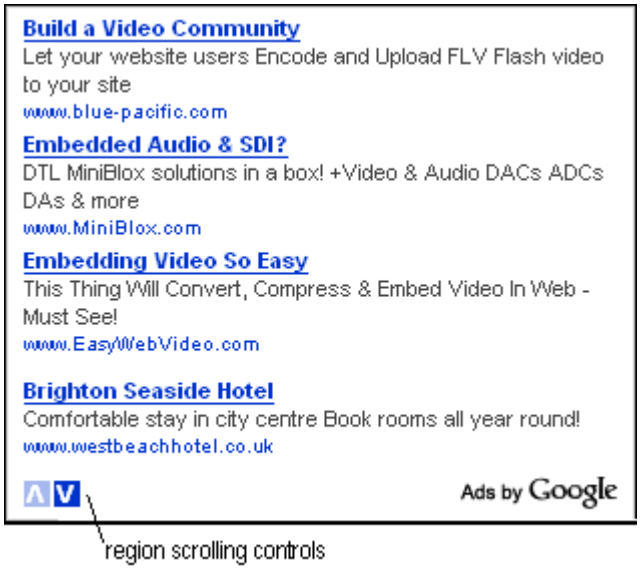


Fig 1.5 b): Un esempio di annunci inseriti dal programma nella pagina web di un publisher

1.2.3 CPA – Cost Per Action (2003)

Questo modello è al momento il più sicuro, ma ancora non viene molto sfruttato. In questo caso il publisher si prende tutti i rischi nel pubblicare l'annuncio pubblicitario, in quanto l'advertiser paga solo quando l'utente, che ha eseguito un click sul messaggio pubblicitario, completa una transazione del tipo fare un acquisto,

registrarsi nel sito o completare un form [POAd]. Il CPC è un caso speciale di CPA in cui l'azione da compiere è quella di cliccare sulla pubblicità. Questo è l'ultimo modello sviluppato cronologicamente parlando che è attivo sul mercato anche se non su ampia scala. Google nel 2007 lo ha provato per un periodo fino ad ottobre 2008 [Wik].

1.2.4 PPPI – Pay Per Percentage of Impression

Questo modello viene qui citato, ma è ancora in fase sperimentale; risulta essere molto robusto riguardo il click fraud e l'impression spam. Il modello prevede che l'advertiser paghi, una cifra pattuita con il publisher, per avere, garantita, una percentuale delle impressioni totali disponibili . Un advertiser chiede che l' *x*% delle volte in cui viene richiesta la tale pagina web, venga impressa la sua pubblicità [PPPe].

Tutto il mondo della pubblicità si basa sulle campagne pubblicitarie; ogni soggetto in campo deve seguire strade appropriate per cercare di ottenere gli obiettivi desiderati.

1.3 Creare e mantenere una campagna pubblicitaria

Per creare una nuova campagna pubblicitaria bisogna prima di tutto scegliere l'ad network a cui ci si vuole appoggiare e fornirle alcuni dati relativi alle credenziali e ad alcune voci finanziarie (come i dati della carta di credito o del conto bancario) per i vari pagamenti. Da questo punto in poi le due parti, advertiser e publisher, seguono due percorsi leggermente diversi.

1.3.1 L'Advertiser

Una volta stabiliti i primi contatti con l'ad network, l'advertiser comincia con il creare diversi esempi di messaggio pubblicitario relativi al suo prodotto e/o servizio variando il tipo, il formato, il colore, il testo etc… Il passo successivo consiste nel dare delle prime indicazioni relative all'indirizzamento della campagna pubblicitaria; per prima cosa si stabilisce un primo periodo di prova in cui mandare in esecuzione tutti gli esempi creati per la campagna. Dopo questo periodo di prova è possibile

analizzare i dati raccolti relativi ai vari messaggi pubblicitari proposti e cercare di migliorare i risultati ottenuti eliminando i messaggi meno proficui e concentrando maggiormente l'attenzione sugli obiettivi desiderati imponendo ulteriori vincoli sui messaggi da riproporre.

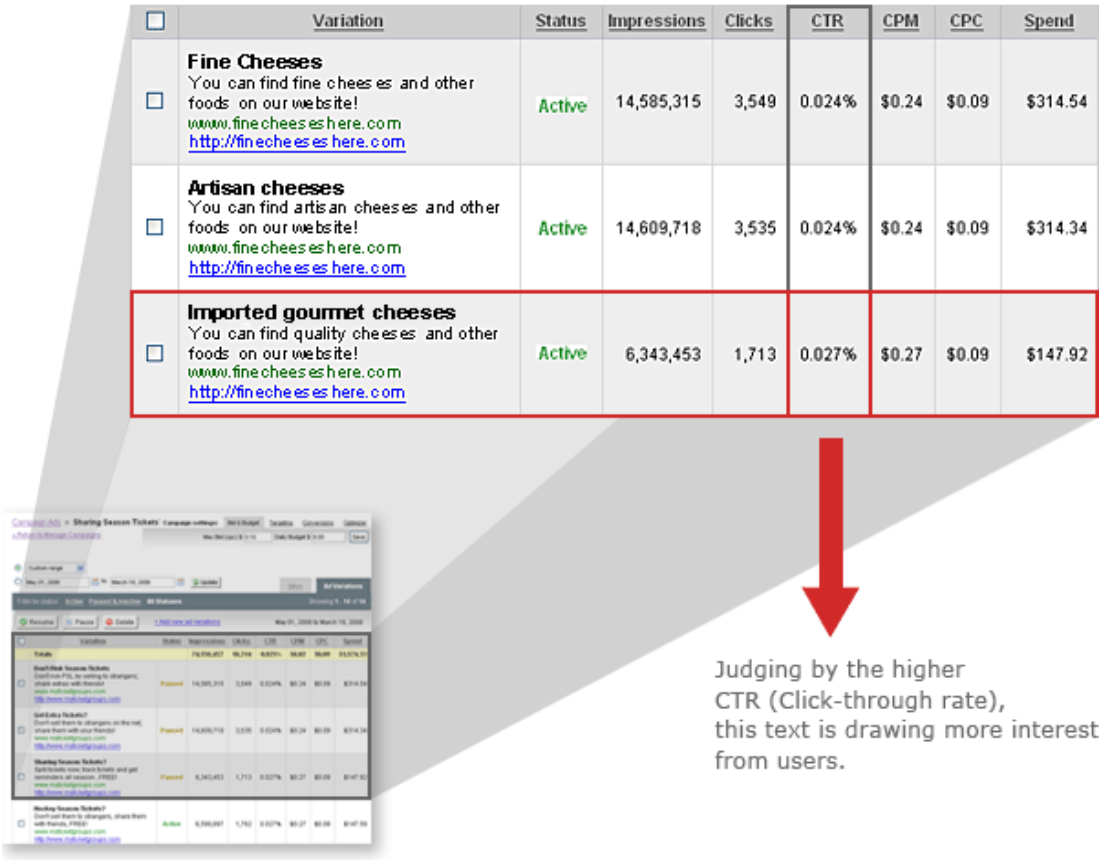


Fig 1.6: Una schermata che raccoglie i dati di una campagna pubblicitaria composta da tre messaggi differenti per lo stesso prodotto [AdBr].

Per migliorare i risultati dei vari annunci bisogna fissare alcuni punti che permettono di caratterizzare in maniera migliore il pubblico a cui è rivolta la campagna pubblicitaria:

- scegliere le parole chiave da associare a ciascun messaggio in modo oculato; per esempio non conviene puntare su parole troppo generali, che sono molto richieste dagli advertiser e quindi più costose per la competizione di altri inserzionisti, conviene scegliere parole che esprimano concetti più precisi, che, anche se sono usate da una minore quantità di pubblico, portano più

clienti interessati al prodotto, permettendo di avere meno concorrenza e meno spese, ma più probabilità di concludere una transazione con l'utente;

- scegliere le categorie dei siti in cui si desidera far apparire i propri annunci, oltre al motore di ricerca associato all'ad network selezionata. Si può quindi decidere in quali categorie di siti comparire in base alla categorizzazione del contenuto delle pagine dei publisher;
- scegliere in quali zone geografiche si preferisce far visualizzare il proprio messaggio ed in quali no. Si può decidere di puntare su una certa regione in cui si ha la maggior concentrazione di clienti, evitando zone in cui, per vari motivi, non c'è la possibilità di avere possibili acquirenti eliminando in tal modo spese inutili;
- specificare per ciascuna parola chiave la "puntata" e il budget giornaliero o settimanale. Si fissa la cifra massima per partecipare all'asta per il ranking dei diversi messaggi e si decide una cifra limite per le spese giornaliere o settimanali in modo tale da evitare, per esempio a causa di click fraud, spese inutili.

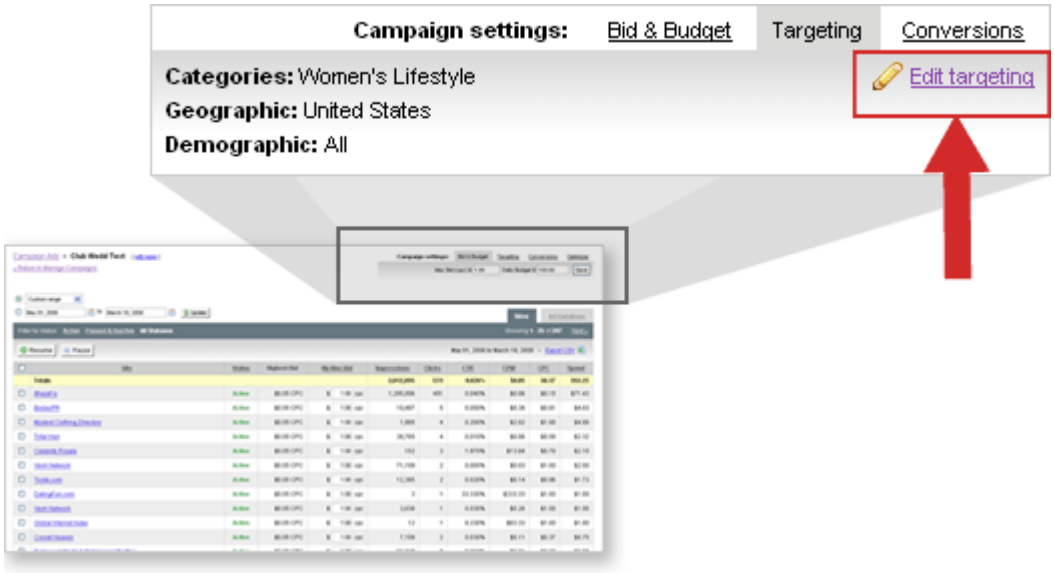


Fig 1.7: Una schermata che mostra i vincoli geografici e di categoria imposti per una campagna pubblicitaria [AdBr].

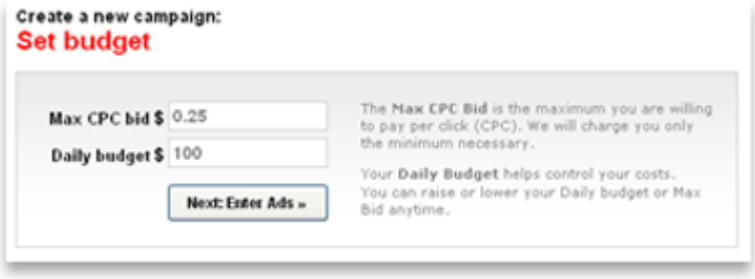


Fig 1.8: Form per l'impostazione della "puntata" massima e del budget giornaliero [AdBr].

Una volta posti questi vincoli, all'advertiser non resta altro da fare che seguire i risultati della sua campagna pubblicitaria, grazie ad alcune applicazioni che ne raccolgono i dati, e sistemare i messaggi tramite opportune modifiche ai punti appena citati o tramite modifiche morfologiche. Se l'advertiser riscontra un ROI positivo significa che la campagna sta andando bene; bisogna comunque continuare a tenerla d'occhio visto che il mercato della pubblicità sul web è in continua evoluzione.

1.3.2 Il Publisher

Dopo aver registrato i propri dati nell'ad network e quindi aver dato la propria disponibilità, il publisher, non ha molto da fare. Anche lui, come l'advertiser, può porre alcune condizioni (poche) per quel che riguarda i messaggi pubblicitari che preferisce visualizzare nelle proprie pagine web:

- decidere il tipo ed il formato (testuali, banner, pop-up, ...) degli annunci pubblicitari visualizzabili sul suo sito e la dimensione massima che questi possono avere;
- decidere gli argomenti dei messaggi da visualizzare; questo si basa sul contesto delle proprie pagine in modo da avere una pubblicità che sia adatta ai contenuti trattati nel proprio sito.

Durante il periodo in cui garantisce la sua disponibilità, il publisher, può variare i punti sopra elencati verificandone i risultati dopo un certo periodo. Grazie ad alcune applicazioni apposite, ne è un esempio AdSense, si possono verificare i ricavi relativi alle pubblicità visualizzate nel proprio sito.

Quando un publisher fissa i punti sopra indicati deve poi permettere che venga inserito all'interno delle proprie pagine il codice Javascript relativo al messaggio pubblicitario da visualizzare; lavorando con un'ad network, l'operazione avviene in maniera automatica.

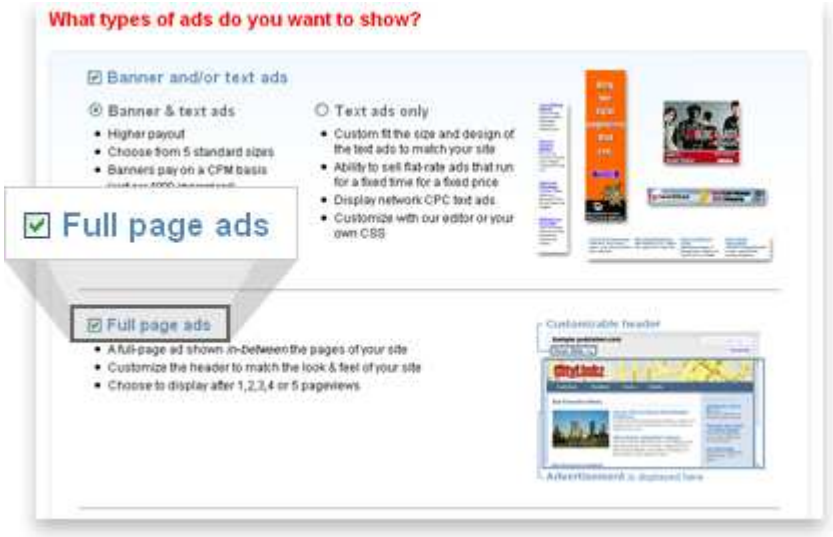


Fig 1.9: Una schermata che permette di scegliere che tipo di messaggi pubblicitari visualizzare nel proprio sito [AdPu].

Questa è stata un'ampia panoramica su quello che è il mondo della pubblicità in Internet, sugli attori in gioco e sulle regole che stanno alla base dei rapporti tra di essi. Nel prossimo capitolo è possibile vedere ed analizzare il problema più grave che affligge il mondo dell'online advertising e cioè il *click fraud* .

2 Click Fraud

Oggi i motori di ricerca sono il primo strumento, per milioni di utenti, per accedere ad innumerevoli informazioni, per navigare i siti web, per cercare notizie e per fare compere online. La maggior parte dei motori di ricerca genera ricavi, tramite la ricerca sponsorizzata, che è diventata parte integrante del loro modello di business. Tutto ciò che può compromettere questo modello ha importanti ripercussioni soprattutto economiche. Il *Click Fraud* è il problema più grosso in questo ambito anche per le sue diverse sfaccettature. Per definizione il click fraud è un crimine che avviene nell'online advertising quando una persona, uno script automatico od un programma imitano il comportamento naturale di un utente cliccando sui messaggi pubblicitari allo scopo di ricavarne un guadagno senza alcun vero interesse per i prodotti e/o servizi offerti dai titolari degli annunci [Wik]. La frode esiste sia nel modello CPM che nel modello CPC. Il click fraud porta sempre ad una riduzione del ROI per gli advertiser. Ci sono anche altri motivi per cui il ROI diminuisce, ad esempio la bassa qualità dei messaggi, la cattiva scelta delle parole chiave, un prezzo di "puntata" non competitivo. Nonostante questo, la frode è una delle variabili da tenere in considerazione in caso di ROI non ottimale da parte dell'advertiser.

Il click fraud consiste, principalmente, nei click intenzionali, fatti su link sponsorizzati, con l'obiettivo di guadagnare soldi in maniera disonesta o per

eliminare la concorrenza. Il click fraud è uno dei problemi che più sta dilagando nel Web; le varie forme con cui si presenta tengono sempre in allarme gli addetti al controllo del traffico web, soprattutto nel caso di traffico non produttivo per l'advertiser. Un click è, per definizione, l'atto di iniziare una visita ad un sito web attraverso un link sponsorizzato o non [ClF], e può essere:

- *valido*; quando è intenzionale ed ha una probabilità reale di generare valore per l'advertiser, una volta raggiunto il sito web;
- *invalido;* quando non c'è possibilità di generare ROI per l'advertiser;

I click invalidi sono a loro volta classificabili in click:

- *fraudolenti*, quando sono fatti con l'intenzione di non generare ricavi per l'advertriser;
- *casuali*, quando non sono fatti maliziosamente, ma sono semplicemente frutto di errori, come ad esempio un doppio click su un link. Questi sono click che un ad network sceglie di non addebitare; il browser dell'utente viene indirizzato comunque al sito dell'advertiser.

Fig 2.1: Una rappresentazione dell'insieme dei click ricevuti da un link e la loro suddivisione in categorie [GoAd].

Il click fraud è, quindi, semplicemente la pratica di commettere click fraudolenti. Siccome l'intenzionalità è solo nella testa delle persone che compiono le azioni o nella mente di chi sviluppa programmi che eseguono click in maniera automatica,è impossibile stabilire con certezza quali click siano fraudolenti. Anche se stabilire in

assoluto l'intenzionalità di un click è impossibile, esistono, però, vari segni o segnali che la possono indicare con vari gradi di certezza. I click che vengono sospettati di essere fraudolenti, vengono segnalati dall'ad network come invalidi. Questo però non garantisce che tutti i click fraudolenti vengano riconosciuti e segnalati come invalidi. Il fatto che il browser utente carichi lo stesso il sito dell'advertiser, anche in caso di click invalido, offre due benefici: per prima cosa un impostore non ha nessun feedback per capire se è stato scoperto oppure no, in secondo luogo se un click sospettato di essere fraudolento è in realtà legittimo (si parla di falso positivo), è permesso lo stesso all'utente di finire la sua transazione andando, così ad incrementare il ROI dell' advertiser. Troppi falsi positivi possono far diminuire i guadagni dei publisher e quindi un ad network deve fare del suo meglio per minimizzarli in modo tale da mantenere un equilibrio tra il garantire alti ROI per gli advertiser e il creare e mantenere relazioni di qualità con i publisher.

Due scenari diversi in cui si verifica il fenomeno del click fraud sono l'*advertiser competitor clicking* e la *publisher click inflation* [OnAF]. Nell'advertiser competitor clicking, un advertiser con cattive intenzioni può cliccare sui messaggi degli advertiser, che sono in competizione con lui per un posto in graduatoria, per tentare di eliminare la concorrenza. Tra le ragioni, per cui un advertiser malintenzionato può tentare la via dell'advertiser competitor clicking, la più importante è senza dubbio la volontà di consumare il budget di un concorrente, eliminandolo così dalla competizione. Un advertiser, quindi, può tentare di far diminuire la competizione per una particolare parola chiave. A causa di questi click fraudolenti, purtroppo, gli advertiser che subiscono ne derivano un minore ROI e vedono sparire i loro budget. Tutto questo fa si che, nel breve termine, le aste diventino meno competitive vista la mancanza di concorrenti eleggibili, mentre nel lungo periodo, a causa della diminuzione del ROI per alcune keyword, gli advertiser concorrenti possano decidere di ridurre o addirittura sospendere le spese riguardo tali parole.

Da quando è stato introdotto da Google il programma AdSense, la publisher click inflation è diventato un nuovo pericolo che necessità di essere limitato. In questa situazione, un publisher esegue dei click sui link presenti nelle proprie pagine, nella speranza di ricevere una parte dei soldi che gli advertiser pagano all'ad network per i

click ricevuti. L'incentivo alla frode è in questo caso molto più diretto al guadagno concreto rispetto al caso dell'advertiser competitor clicking.
Spesso i publisher, invece di compiere in prima persona la frode, chiedono ai loro utenti, ignari, di compiere dei click su alcuni link, con motivazioni fasulle come 'supportare il loro sito', oppure, attraverso degli script nascosti nelle proprie pagine, fanno eseguire ai browser degli utenti dei click nascosti. Nei web server viene tenuta traccia di tutti i click che vengono fatti registrando le varie richieste http ricevute; in questo modo è possibile analizzare, poi, il traffico web alla ricerca di dati anomali. Chi cerca di fare click abusivi finisce sempre con l'incrementare il CTR di certe pagine, per cui rapporti più alti del normale possono servire come sentinelle di allarme per quel che riguarda un attacco di click fraud.

2.1 Gli Attacchi

La pratica del click fraud può avvenire in molti modi e secondo diverse forme. C'è chi cerca semplici attacchi da casa e chi si dedica a queste attività per lavoro arrivando a scrivere dei malware per infettare macchine in tutto il mondo. Lo spazio degli attacchi alle reti si divide in due categorie principali: *attacchi manuali*, in cui le persone manualmente vanno a cliccare sugli annunci pubblicitari; e *attacchi automatizzati*, in cui il traffico è generato da software programmabili.

2.1.1 Attacchi manuali

Per attacchi manuali si intendono tutte quelle forme di attacco che prevedono il coinvolgimento di una persona nel ruolo determinante di esecutore del click fraudolento[OnAF]. Le persone sono difficili da gestire rispetto ai software automatizzati, inoltre sono limitate fortemente dalla velocità con cui riescono a pensare ed agire. Nonostante questo, assumere persone per eseguire click su link sponsorizzati è una pratica molto in uso per attaccare un ad network in paesi in cui le persone lavorano per cifre veramente modeste; per i publisher, che ricevono in cambio un guadagno per i click fatti dal proprio sito, questi costi sono comunque abbastanza bassi rispetto ai ricavi [OnAF]. I malintenzionati possono generare diversi attacchi che sfruttano le prestazioni umane; questi attacchi hanno tutti

obiettivi simili, ma hanno diversi costi e differenti gradi di difficoltà nell'implementazione. Gli attacchi manuali possono prevedere il coinvolgimento di una singola persona, oppure il coinvolgimento di più persone il cui lavoro può essere coordinato. Un modo per nascondere questi attacchi è quello di usare un ampio numero di macchine diverse in modo tale da avere a disposizione diversi indirizzi IP.
Un sito ben costruito può stimolare l'utente a compiere certe azioni; è per questo che certe volte lo scopo di alcuni web-designer è quello di creare siti che "incoraggino" i visitatori a cliccare sugli annunci CPC che sono presenti nelle pagine.
Per essere sicuro di presentare nelle proprie pagine link che portino buoni guadagni, un publisher può rimpiazzare, o semplicemente aggiungere, al regolare contesto della pagina, alcune keyword che conosce essere molto proficue. Questa tecnica è detta "*keyword stuffing*" [OnAF]. Le parole chiave scelte, possono essere nascoste all'utente inserendole nei tag HTML che non vengono visualizzati, o scrivendo le parole con lo stesso colore dello sfondo della pagina. In ogni caso le parole così inserite vengono riconosciute dall'ad network, quando analizza le pagine per determinare il contesto in base al quale scegliere le pubblicità rilevanti da mostrare, ma non vengono viste dagli utenti che visitano le pagine. In maniera simile un publisher può modificare il testo del messaggio per renderlo più interessante e convincente per gli utenti, mentre il link rimane lo stesso. In questi casi il messaggio varia in maniera totale passando addirittura da un argomento (ad esempio viagra) ad un altro (ad esempio suonerie gratis) [OnAF].
Per incoraggiare i visitatori a seguire un link CPC di alto rendimento, i publisher non includono molti link 'regolari' ad altre pagine; per cui un utente che vuole andare ad un'altra pagina è quasi costretto a cliccare su uno dei link CPC.
Un publisher molto subdolo può arrivare a rendere invisibile l'annuncio pubblicitario all'utente e dargli delle indicazioni per quanto riguarda le azioni da compiere per forzarlo ad eseguire un click sul messaggio senza che se ne renda conto (ad esempio "premi due volte tab e poi entra").
Un ultimo strattagemma per ingannare gli utenti è quello di ospitare nelle proprie pagine dei giochi, spesso scritti con Macromedia Flash, che sono legati a della pubblicità; parte del gioco può richiedere di eseguire dei click o di muovere il mouse;

in questo modo, spesso, i visitatori tendono a cliccare inavvertitamente sulle pubblicità garantendo un profitto al publisher.

Per quanto riguarda la competizione tra advertiser, un inserzionista che vuole eliminare un concorrente consumando il suo budget lo può fare giorno per giorno se non vengono prese le appropriate contromisure. D'altro canto questo comportamento può essere facilmente identificato se lo stesso utente ritorna nello stesso sito più e più volte in un breve periodo.

Per rendere un attacco manuale difficile da riconoscere alcuni attaccanti usano i *proxy http* per oscurare la vera sorgente dei click. I proxy http possono rendere anonimo il traffico agendo come intermediari tra la macchina dell'utente ed il server web dell'advertiser; i proxy riescono a nascondere l'indirizzo IP sorgente e a distruggere le informazioni che riguardano l'identità dell'utente, come i cookies, nelle richieste http.

Il vantaggio principale, nell'usare persone nel cercare di compiere click fraud è soprattutto economico, mentre gli apetti negativi sono rappresentati dal fatto che le persone sono difficili da convincere e da incentivare a lavorare e possono annoiarsi e stancarsi facilmente. Va aggiunto che una persona pagata per guardare pagine web e cliccare sugli annunci pubblicitari agisce in maniera totalmente diversa rispetto ad un utente veramente interessato al prodotto; in quanto, per esempio, un reale acquirente è portato a leggere le pagine web, considerare gli argomenti trattati, pensare ad eventuali azioni da compiere e navigare nel sito per conoscere il prodotto, prima di agire, mentre una persona non interessata arriva subito a cliccare link, per cui le sue richieste http sono molto ravvicinate nel tempo e quindi sentore di frode.

2.1.2 Attacchi automatizzati

Con il termine attacchi automatizzati si intendono tutte le forme di attacco che prevedono l'utilizzo di strumenti che svolgono l'attacco in maniera autonoma e automatica all'insaputa dell'utente [OnAF]. Spesso coloro che vogliono compiere un attacco abbandonano ogni pretesa di legittimità nei loro sforzi per cui ricorrono al traffico automatizzato generato da software detti "*bots*". Questi programmi hanno un grande vantaggio sugli utenti umani; loro fanno quello che gli viene ordinato in continuazione e lo fanno gratis, senza che ci sia bisogno di motivazioni. Far lavorare

un software richiede meno coordinazione ed evita all'attaccante il fastidio di dover cercare di interagire con delle persone. Alcuni software nascono con scopi legittimi come lo scrapping dei siti web o il crawling dei link, mentre altri vengono costruiti al solo scopo di eseguire dei click sui messaggi pubblicitari. I software nati con lo scopo di cliccare sulle pubblicità vengono detti *clickbot* [OnAF]; questi programmi fanno tali operazioni rilasciando delle richieste http per le pagine web degli advertiser con l'intento di commettere click fraud. Questo tipo di programmi possono essere creati su misura oppure acquistati, possono essere installati manualmente dagli impostori oppure diffusi in maniera simile ai malware ed ai worms. Sfortunatamente per questi clickbot hanno la debolezza di avere un comportamento spesso prevedibile.

Sviluppare un programma che si comporti come una persona e riesca a nascondere il suo comportamento è molto difficile. Un'ad network può usare centinaia o migliaia di segnali per cercare di capire se una richiesta http viene generata da una macchina in maniera autonoma o se invece è generata da un utente. Nonostante ciò molte sorgenti di traffico automatizzato vengono usate dagli attaccanti per spedire richieste non proficue per gli advertiser. Costruire un programma che esegua i click sui messaggi pubblicitari richiede alcune abilità come programmatore e richiede anche una profonda conoscenza su come poter scrivere un software che funzioni come se fosse umano. Generalmente esistono diversi tipi di software maligno:

- software creati su misura; ci sono molti strumenti utili per sviluppare questi software, alcuni sono addirittura open-source;
- software per la vendita; quando è finito un programma può essere venduto nel mercato, esistono applicazioni legittime poi acquistate per usarle in modo maligno. Esempi di programmi acquistabili per tentare il click fraud sono I-Faker, FakeZilla e Clickmaster; essi usano tipicamente i proxy per avere un traffico con diversi IP; fortunatamente la diversità degli IP non è sufficiente per nascondere gli attacchi condotti da questi software;
- software di tipo malware; questi programmi infettano le macchine per avere così diversi indirizzi IP a disposizione per gli attacchi; il loro traffico può

essere più o meno facile da identificare rispetto a quello dei software precedenti.

Esiste poi una tecnica particolare che prevede di “forzare” dei click da parte del browser utente. Con questa tecnica l’attaccante può tentare di convertire il browser dell’utente in un agente che funge da clickbot. La tecnica sfrutta a proprio vantaggio alcuni difetti di alcune implementazioni di programmi, simili ad AdSense, che possono essere usati da publisher malintenzionati per compiere attacchi del tipo click inflation. Da questi programmi i publisher ricevono un pezzetto di codice HTML che inseriscono nelle loro pagine web nelle posizioni in cui vogliono far apparire le pubblicità; un publisher può, a questo punto, inserire nel codice HTML addizionale alcune istruzioni che istruiscono il browser utente in modo tale che quest’ultimo clicchi sui link presenti nella pagina senza che ci sia una richiesta esplicita dell’utente stesso.

2.1.3 Un attacco indecifrabile

Sempre parlando di attacchi automatizzati, esiste un attacco di tipo click inflation che è tutt’ora difficile, se non impossibile rilevare. Questo attacco è molto efficace perché permette ad un publisher di aumentare i propri guadagni ed allo stesso tempo non viene riconosciuto dall’advertiser. In questo caso è previsto che il publisher sia associato ad altri siti tramite un accordo per cui visitando le pagine web di tali siti si apre la pagina del publisher che esegue un click nascosto, in maniera automatica, verso l’advertiser. Vediamo un esempio concreto in cui il sito *S* si è accordato con il publisher *R*. Quando un utente entra nella pagina web di *S* causa un click simulato verso il sito di *R*, il click è invisibile all’utente. Questa operazione prevede l’invio da parte del browser utente di una richiesta http per il sito di *R* con referente il sito di *S*. Come risposta a questa richiesta, che ha per referente *S*, *R* ritorna una versione modificata della sua pagina web al browser che causa un click nascosto verso la pagina dell’advertiser *T*. Questo click nascosto, crea unaa richiesta http del browser per la pagina di *T* con referente il sito di *R*; perciò *T* paga *R* come publisher referente. Nel caso in cui una richiesta della pagina di *R* non abbia come referente *S* o un altro sito associato, viene caricata la pagina “normale”, “innocua”, di *R* che non simula nessun click verso *T*. Per questo se un webmaster di *T* vuole controllare le pagine di

R, quando carica le pagine non trova nessuna evidenza del gioco sporco fatto da *R* in quanto le sue richieste non hanno come referente *S* e T non è associato a R [SecP]. Le pagine di *R* sono pagine che hanno una doppia personalità, "buona" quando non simula click e "cattiva" quando ha come referente *S* o un altro sito associato [Bad].

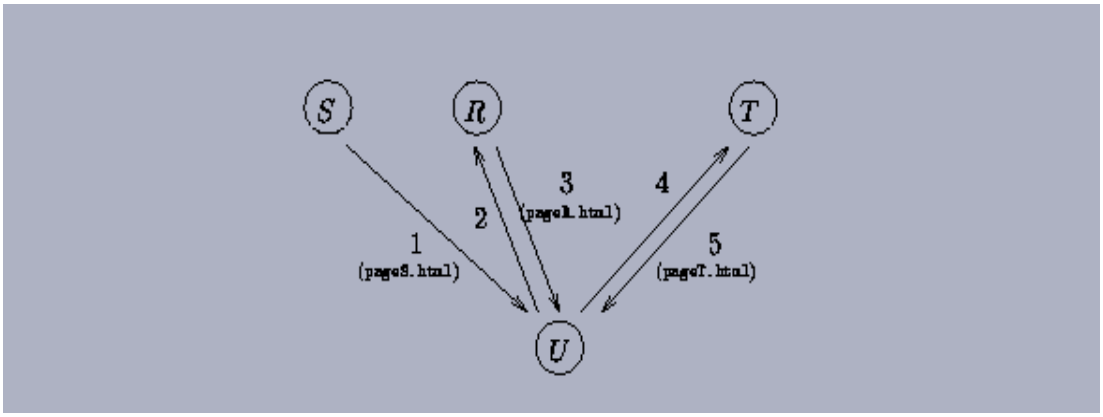


Fig 2.2: Una rappresentazione dell'attacco: 1) richiesta dell'utente U di una pagina di S, 2) click invisibile verso R con referente S, 3) caricamento della versione modificata di R, 4) click invisibile verso T con referente R, 5) caricamento della pagina di T che paga R [SecP].

Questo attacco, come visto, si basa sulla realizzazione di due click simulati in maniera da non essere visibili all'utente. Una caratteristica che rende tutto questo possibile è che i browser moderni trasmettono le informazioni di referenza non solo per le pagine richieste esplicitamente dall'utente, ma anche per i componenti inclusi nelle pagine come immagini e sottodocumenti. Spesso per generare i click simulati viene usato Java Script. Quando uno script Java Script causa un click simulato il browser si comporta come se l'utente avesse effettivamente cliccato sul link. Per nascondere il click all'utente il referente può fare in modo che il contenuto dell'URL cliccato venga caricato in una finestra separata nascosta sotto la finestra principale del browser utente; a questo punto il referente può fare in modo che la nuova finestra venga chiusa velocemente quando è cominciato il caricamento della pagina successiva [SecP]. Un utente attento potrebbe notare la comparsa di questa finestra sulla toolbar del desktop per un istante, ma nella maggior parte dei casi non viene notata.

2.2 Le Contromisure

Per cercare di individuare e frenare il fenomeno del click fraud esistono alcune contromisure che possono essere usate da advertiser, publisher ed ad network. Molte contromisure e difese vengono usate per tentare di ridurre la probabilità di successo di un attacco. Le difese possibili sono suddivisibili per tipo di metodologia:

- prevenzione;
- riconoscimento;
- contenimento.

Con queste difese non c'è nessuna pretesa di eliminare in assoluto il fenomeno del click fraud; il loro scopo è, per l'appunto, quello di abbassare in maniera decisa la probabilità di successo di un attacco e di supportare un sistema proficuo per tutte le componenti coinvolte. Le contromisure, qui presentate, non sono necessariamente indirizzate al riconoscimento dei click fraudolenti; ma sono focalizzate verso l'individuazione di fenomeni anomali e comportamenti inusuali o di altre inconsistenze che possono essere indicative di una frode. In molti casi, i click associati con queste anomalie vengono segnalati come invalidi e non prevedono nessun pagamento.

2.2.1 La Prevenzione

Mentre è impossibile prevenire il comportamento di qualcuno che manualmente clicca sui link, è possibile prendere delle precauzioni che possono prevenire attacchi su larga scala e attacchi sistematici. Nel caso di attacchi dovuti al click inflation del publisher, questa frode può essere limitata facendo attenzione a quali publisher ammettere nei programmi come AdSense. I publisher che sono stati eliminati per la loro "bassa qualità" o per click fraudolenti cercano, a volte, di ripresentarsi con nomi (falsi) diversi, indirizzi nuovi e altri recapiti per essere riammessi nel programma. Per evitare sbagli è importante, per un ad network, identificare univocamente i publisher per poterli sempre riconoscere.

Le ad network possono vedere e operare su diverse caratteristiche delle richieste http che possono fornire indicazioni sulle attività fraudolente. Se un attaccante conosce quali sono le caratteristiche analizzate da un ad network per suddividere i click, può

costruire artificialmente richieste http che non esibiscano caratteristiche anomale. Mentre un ad network può essere abbastanza trasparente circa le contromisure e i processi in uso, è estremamente importante che mantenga la confidenzialità per ciò che riguarda i segnali che usa per individuare i click invalidi.
Un ultimo, ma non meno importante, suggerimento per prevenire il click fraud è quello di fissare, da parte degli advertiser, un massimo CPC che permette di evitare che un attaccante riesca a guadagnare molto con pochi click od a danneggiare in maniera pesante l'advertiser stesso. Grazie a questa accortezza gli attaccanti che vogliono ricavare una cifra sostanziosa sono costretti a produrre un traffico pesante che può essere interpretato correttamente come un segnale significativo di attività anomala.

2.2.2 Il Riconoscimento

Lo scopo del processo di riconoscimento è quello di identificare i click invalidi una volta che questi sono stati praticati. L'operazione di riconoscimento dei click può essere fatta sia "online" che "offline" [OnAF]. Quando i click non validi vengono riconosciuti online gli advertiser non devono pagare, mentre quando vengono identificati offline gli advertiser vengono rimborsati, oppure gli vengono accreditati dei click.
L'identificazione online dei click è preferibile, per gli advertiser, in quanto il loro budget giornaliero non viene, così, toccato dai click invalidi, e gl'advertiser stessi non perdono l'opportunità di continuare a partecipare alle aste per gli slot delle pagine in cui compare la pubblicità.
Quando i click non validi vengono identificati offline, gli advertiser vengono accreditati di click da consumare in futuro oppure vengono rimborsati. In questi casi l'advertiser può trovarsi a non competere più per un posto nella pagina in quanto il suo budget giornaliero viene consumato da un attacco di click fraud. Bisogna sottolineare che alcuni tipi di attacchi possono essere più facili da individuare offline perché richiedono l'analisi di una grossa quantità di dati; ad esempio un attacco costituito da pochi click ogni giorno richiede molti giorni di traffico aggregato per l'identificazione. Questi tipi di attacco sono difficili da riconoscere online quando avvengono, soprattutto nella loro parte iniziale.

I metodi per il riconoscimento del click fraud normalmente si basano sull'identificazione di anomalie all'interno di grosse serie di dati. Diverse anomalie possono presentarsi in diversi momenti nell'analisi dei click. Alcune anomalie possono apparire evidenti in un singolo click che può essere segnalato come invalido da un filtro online; altri tipi di anomalie, invece, possono presentarsi in maniera chiara solo nell'analisi di traffico aggregato relativo ad un certo periodo di tempo. Per quanto riguarda la decisione su quando usare un filtro online od offline per i click, spesso la scelta dipende dal tipo di anomalie che si vuole scoprire e dal periodo di tempo che si vuole analizzare per identificare tali anomalie.

Dopo che i filtri sono stati applicati, e una volta che i click invalidi sono stati segnalati vengono intraprese altre operazioni; peer esempio, un publisher di AdSense con un'alta percentuale di click invalidi, o anche uno che riceve molto traffico potenzialmente anomalo che può o no essere invalido, può essere fatto ispezionare da un investigatore della qualità del traffico [OnAF]. In alcuni casi, una relazione con un publisher può terminare automaticamente se una proporzione smisurata di click non validi proviene dal suo sito.

Ci sono situazioni in cui ci sono talmente tanti dati di click associati ad un publisher od ad un advertiser che può essere difficile individuare i click fraudolenti perché sono, letteralmente, "persi nel mare" dei click validi. In questi casi il danno economico dovuto al click fraud può essere relativamente basso se comparato con il numero aggregato dei click. Nonostante questo, identificare la dimensione giusta per suddividere i dati in blocchi analizzabili accuratamente è un aspetto importante del problema.

In aggiunta all'identificazione"passiva" delle anomalie nei dati, un ad network può condurre un riconoscimento "attivo" modificando l'interazione con il browser o con l'utente per generare dati aggiuntivi che possono essere usati per accertare la validità di un click. Un ad network può decidere di aggiungere delle variabili Java Script, allo script inserito negli slot pubblicitari, che eseguano un'operazione nei browser dei clienti; una volta ricevuto un click, l'ad network, può vedere in queste variabili i risultati dell'operazione eseguita e se sono compatibili con i risultati attesi verificare la validità del click.

2.2.3 Il Contenimento

Le ad network di più alto livello utilizzano centinaia o migliaia di segnali, attivi o passivi, per combattere il click fraud. Nonostante ciò, tipicamente è difficile stabilire con assoluta certezza la validità di un click. Per questo, le ad network più prudenti hanno stabilito che alcuni tentativi di frode non possono essere direttamente riconosciuti ed eliminati, per cui è importante contenere ed amministrare il potenziale impatto economico di questi attacchi per gli advertiser. Un primo modo molto semplice di limitare l'impatto degli attacchi che provengono da uno stesso IP consiste nell'inserire uno script, scritto dai programmatori, che mostra un messaggio di ammonimento dopo che sono arrivate ripetute richieste http per il sito dell'advertiser, provenienti dallo stesso IP tramite i link a pagamento di alcune pagine dei publisher.

Fig 2.3: Un esempio di messaggio di ammonimento per dei click ricevuti dallo stesso indirizzo IP [Clab].

Esistono poi, principalmente, altre due misure di contenimento: lo smart pricing ed le manual reviews. A causa della difficoltà nell'accertare quando un click è valido o no, alcune ad network hanno pensato, per alcuni click, di far pagare agli advertiser solo una frazione del CPC pattuito. La versione di Google di questo sistema si chiama *smart pricing* [GoWo]. Basandosi su vari fattori che possono essere indicativi del ROI degli advertiser, come CTR e CPC medio, la qualità dell'intero traffico può

essere controllata e catturata. Questa strategia di pagamento può proteggere gli advertiser dal pagare l'intero CPC per click che poi non si convertono in un guadagno per loro; in più può contenere gli effetti della scarsa qualità di alcune pubblicità.

Le *manual reviews* (revisioni manuali) [OnAF] del traffico dei click possono essere anch'esse una parte importante della strategia di contenimento di un ad network. Se alcuni click invalidi superano i meccanismi automatici di identificazione delle frodi, possono essere rilevati da ingegneri altamente istruiti e personale specializzato che con il tempo hanno sviluppato un acuto "olfatto" e un ottimo intuito. In alcune situazioni, le richieste di revisioni manuali (generate da processi interni o dagli advertiser) possono portare alla scoperta di nuovi tipi di attacco; in questi casi, gli ingegneri ed il personale operativo possono provare a generalizzare il nuovo attacco per crearne una descrizione in base alla quale realizzare delle contromisure automatizzate per il suo riconoscimento futuro.

I feedback degli advertiser vengono presi in considerazione in quanto possono, qualche volta, scoprire dei click invalidi non rilevati a causa di errori nel sistema di riconoscimento; per questo i feedback degli advertiser più importanti e di migliore qualità possono essere usati per garantire un miglior ROI a tutti gli advertiser e benefici per l'ad network stessa. La maggior parte delle indagini dei singoli advertiser finisce, comunque, non scoprendo click invalidi non rilevati, ma, invece, scoprendo dettagli relativi alla non ottimalità della gestione della propria campagna pubblicitaria o del design del proprio sito. In pochissimi casi, gli advertiser identificano correttamente i click fraudolenti che già sono stati rilevati dal sistema dell'ad network e per cui non hanno pagato; in ancora meno casi aiutano a scoprire click fraud non rilevati.

Google riporta che meno del 10% dei click vengono considerati giustamente invalidi; questi includono sia i click fraudolenti che quelli ridondanti o casuali e gli advertiser non pagano per questi click. Tutti questi click vengono considerati proattivamente invalidi da Google [Off07].

Queste metodologie vengono usate dalle ad network per gestire il traffico di click tra i publisher e gli advertiser che fanno riferimento a loro. Se un advertiser vuole

controllare da se il traffico in entrata al suo sito da publisher esterni lo può fare accordandosi con personale specializzato nell'analisi del traffico di siti web o usando dei tools appositi per analizzare i click; si parla di *click fraud auditing*.

2.2.4 Click Fraud Auditing

Il click fraud auditing è il processo che consiste nell'analizzare i click indirizzati verso il sito dell'advertiser che provengono da altri siti per determinare quali click, se ce ne sono, sono fraudolenti e/o fatti con intenzioni cattive. Mentre un ad network può analizzare il traffico dei click come parte integrante del processo di riconoscimento del click fraud, una compagnia che lavora come terza parte (diversa dall'ad network e dall'advertiser) può fare solo un esame sommario.
In questi casi un advertiser è interessato a determinare se un ad network sta riportando correttamente il numero di click validi per i quali deve pagare. L'advertiser può assumere un'organizzazione di terza parte per fungere da *auditor* [OnAF]. L'ad network e l'auditor possono lavorare assieme per cercare di eliminare le discrepanze tra il numero di click validi ed invalidi indicati nei log dell'ad network e nei log dell'advertiser. I rapporti degli auditor, quando sono costruiti correttamente, hanno il potenziale per:

- garantire i ricavi dell'advertiser dandogli la sicurezza del fatto che sta pagando per le visite effettivamente ricevute;
- dare peso agli auditor come terze parti fidate che possono aiutare a risolvere le discrepanze;
- aiutare le ad network ad identificare e fissare i limiti nei loro sistemi di riconoscimento del click fraud.

Quando, questi rapporti, sono fatti male, possono dare delle false informazioni agli advertiser e causare delle modifiche alle campagne pubblicitarie dagli effetti economici e finanziari negativi. Le compagnie che fungono da auditor sono limitate per quanto riguarda i dati a disposizione per decidere la validità dei click; esse, infatti, possono ricavare informazioni solo dalle richieste http che arrivano al sito dell'advertiser. Visto che le ad network non possono rendere pubbliche, chiaramente, le informazioni su quali sono esattamente i click ritenuti validi e quali no, una

compagnia di auditing è obbligata a cercare di scommettere su quali siano i click considerati invalidi dall'ad network. Una compagnia di auditing può cercare di fare le proprie scelte in base alle anomalie riscontrate. Per dare un peso alle diverse anomalie si può procedere con un'analisi che si compone di diversi passi; ad ogni passo si da un peso e alla fine si ha un punteggio che indica la qualità delle visite al sito dell'advertiser (passi eseguiti da Clicklab nei suoi tools)[Clab]:

1. visita in profondità: controllare quante pagine del sito sono state generate nella sessione considerata; se risulta una pagina soltanto c'è una buona ragiona per avere dei sospetti. Bisogna anche però ricordare che le cause di una visita così breve possono essere molte, dovute anche alla qualità;
2. IP dei visitatori: controllare che non ci siano troppe visite dovute allo stesso IP; se lo stesso IP supera una soglia di visite in un breve periodo di tempo viene segnalato come anomalia;
3. IP dei click a pagamento: in maniera simile alla precedente, si analizzano solo le sessioni provenienti dai link a pagamento presenti in altre pagine web; si può vedere l'indirizzo di destinazione per capirne la sorgente (ad esempio advertsito.com/?source=google);
4. presenza di cookie: una sessione senza cookie è un buon segnale per dare l'allarme; bisogna anche considerare il fatto che spesso, per la privacy, i cookie non vengono accettati, ma non vuol dire che ci siano cattive intenzioni;
5. frequenza delle pagine: molti programmi, che navigano un sito, richiedono le pagine al server molto più velocemente di un utente normale. Se una sessione genera alcune pagine in pochissimi secondi è giusto penalizzarla; bisogna fare attenzione alla soglia temporale che si impone;
6. proxy server anonimi: gli indirizzi IP sono i primi segnali di attacco, infatti, gli attaccanti spesso sfruttano i proxy server anonimi per avere a disposizione differenti IP. Una soluzione, per intercettare questi

attacchi, è quella di sviluppare e mantenere una lista dei proxy server esistenti e penalizzare le sessioni provenienti da loro;

7. origine geografica: si può tenere una lista nera dei paesi che non sono interessanti per l'advertiser; in questo modo non si impedisce loro di accedere al sito, ma si considerano come fonti probabili di frode.
8. orari privilegiati: se la maggior parte dell'attività di business si svolge in un certo periodo della giornata, si può decidere di penalizzare i visitatori che generano delle sessioni fuori da questa fascia oraria.

Stabiliti questi punti è necessario, prima di agire, fare un meeting per decidere le soglie da impostare sui vari test e i punteggi da assegnare nei vari casi. Una volta fatto questo si prova il sistema, prima con degli ipotetici utenti reali per verificare che il loro punteggio non generi l'allarme, e poi con delle visite fraudolenti fatte appositamente per verificare che in ogni caso venga dato l'allarme.
I rapporti generati da queste compagnie di auditing aiutano gli advertiser, che controllando attentamente il loro ROI, possono sistemare in maniera appropriata le loro campagne pubblicitarie. Il click fraud non rilevato si manifesta con un abbassamento del ROI rispetto al previsto (per esempio rispetto al passato). Allo stesso tempo molti altri fattori come la scelta non ottimale delle keyword o delle "puntate", o un concorrente con un miglior prezzo possono impattare negativamente sul ROI. Quindi, mentre un click fraud non riconosciuto abbassa sicuramente il ROI, un ROI basso non significa necessariamente che una frode non segnalata sia in atto.
Si potrebbe pensare che se le ad network hanno molti più dati delle compagnie di auditing, allora forse le stesse ad network potrebbero condividerli con le compagnie per permettere a loro di aiutare nella verifica dei click. Sfortunatamente, per proteggere la privacy dei suoi utenti, un ad network non può condividere grossa parte dei dati che sarebbero necessari per queste verifiche.

In questo capitolo sono stati discussi molti punti per quel che riguarda il problema del click fraud nell'ambito dell'online advertising. Ci sono molti aspetti che riguardano la mitigazione di questo fenomeno che sono ancora oggetto di ricerca perché questo è un mercato in continua espansione. Una gestione efficiente delle

frodi permette, comunque, vantaggi competitivi alle ad network e le aiuta nel garantire agli advertiser il ROI più alto possibile.

3 Catalogo: l'OPAC dell'ateneo

Questo capitolo vuole presentare la struttura e le funzioni del portale Catalogo del Sistema Bibliotecario Padovano relativo all'OPAC (Online Public Access Catalogue, Catalogo in linea di pubblico accesso) delle biblioteche di ateneo e di quelle convenzionate.

Da una rapida interrogazione su un motore di ricerca si scopre che il catalogo è raggiungibile direttamente all'indirizzo: http://catalogo.unipd.it/F?func=find-b-0.

Questo catalogo permette di verificare se un documento, in qualsiasi formato, è posseduto da una o più biblioteche.

La prima pagina che si incontra nel sito è quella relativa alla Ricerca semplice. Le possibilità date dal menù principale sono:

- Ricerca semplice (pagina attuale);
- Altre ricerche;
- Elabora ricerca;
- Ricerche eseguite;
- La mia cartella;

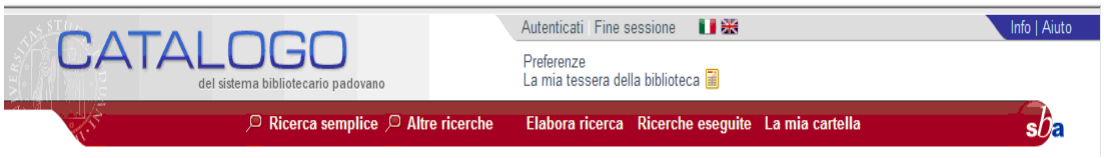


Fig 3.1: Menù principale del catalogo del sistema bibliotecario padovano [Cat].

Come si vede dall'immagine il catalogo può essere visualizzato anche in inglese; inoltre, se l'utente è uno studente universitario o è già registrato presso le biblioteche convenzionate, può autenticarsi per vedere le sue ricerche precedenti e i suoi dati relativi all'utilizzo del catalogo ed eventuali prestiti.
Bisogna sottolineare che per consultare il catalogo non è necessario autenticarsi.

3.1 Ricerca semplice

Questa prima possibilità di ricerca offre in realtà diverse alternative. La pagina presenta una casella di testo in cui inserire la query di interesse, poi due menù; il primo menù permette di stabilire a che parametro si riferisce la query inserita, mentre il secondo menù serve per selezionare il tipo di materiale più di interesse.
Dal primo menù si vede come sia possibile riferire la query a diversi campi di ricerca. La stringa inserita può essere ricercata come:

- Parola chiave: cerca la stringa come termine importante nel contesto complessivo dell'opera cioè tra gli autori, il titolo ed il soggetto;
- Parole del Titolo: cerca la stringa tra le parole del titolo delle varie opere;
- Scorri Titolo(senza articolo): cerca la stringa nel titolo delle opere proponendole in ordine di posizione della stringa nel titolo senza considerare gli articoli iniziali, se la stringa inserita è la prima parte di un titolo, quel titolo verrà considerato rilevante nei risultati rispetto a opere il cui titolo contiene la stessa stringa nella parte finale;
- Autore: cerca la stringa nei nomi degli autori;
- Soggetto: cerca la stringa negli argomenti trattati dall'opera.

Fig 3.2: Parte centrale della pagina per la Ricerca semplice[Cat].

Come accennato il secondo menù permette di scegliere il materiale di interesse tra i diversi tipi disponibili:

tutti, grafica, libro, libro antico, libro elettronico, mappa, multimedia, musica a stampa, registrazione sonora, risorsa elettronica, rivista, rivista elettronica, spoglio, video.

Dalla guida e dagli esempi visti nel sito è indifferente utilizzare lettere minuscole o maiuscole, si possono, inoltre, utilizzare gli operatori booleani AND, OR, e NOT nella stringa di ricerca [Ca_Gu].

Il link , nell'immagine, in basso a destra, permette di avere ulteriori opzioni di ricerca per quanto riguarda il campo da ricercare, rendendo disponibili altri tipi di ricerca quali le ricerche per Identificativo (ISSN, ISBN, BID, …) e le ricerche per Classificazione (Dewey, CDU, MSC, …). Il link offre anche la possibilità di filtrare i risultati della ricerca per lingua, anno di pubblicazione e biblioteca o polo di appartenenza.

3.2 Altre ricerche

Attraverso la voce Altre ricerche nel menù orizzontale si entra in una pagina che permette di eseguire una ricerca più approfondita e dettagliata richiedendo l'inserimento di diversi dati relativi alle opere di interesse per l'utente.

La pagina propone un menù a linguette con diverse opzioni:

3.2.1 Campi

Questa linguetta permette di inserire dati precisi per quel che riguarda l'opera o le opere di interesse per l'utente. In questo caso è possibile inserire l'autore, il titolo esatto, altre parole del titolo (se non si conosce esattamente), anno di pubblicazione, editore.

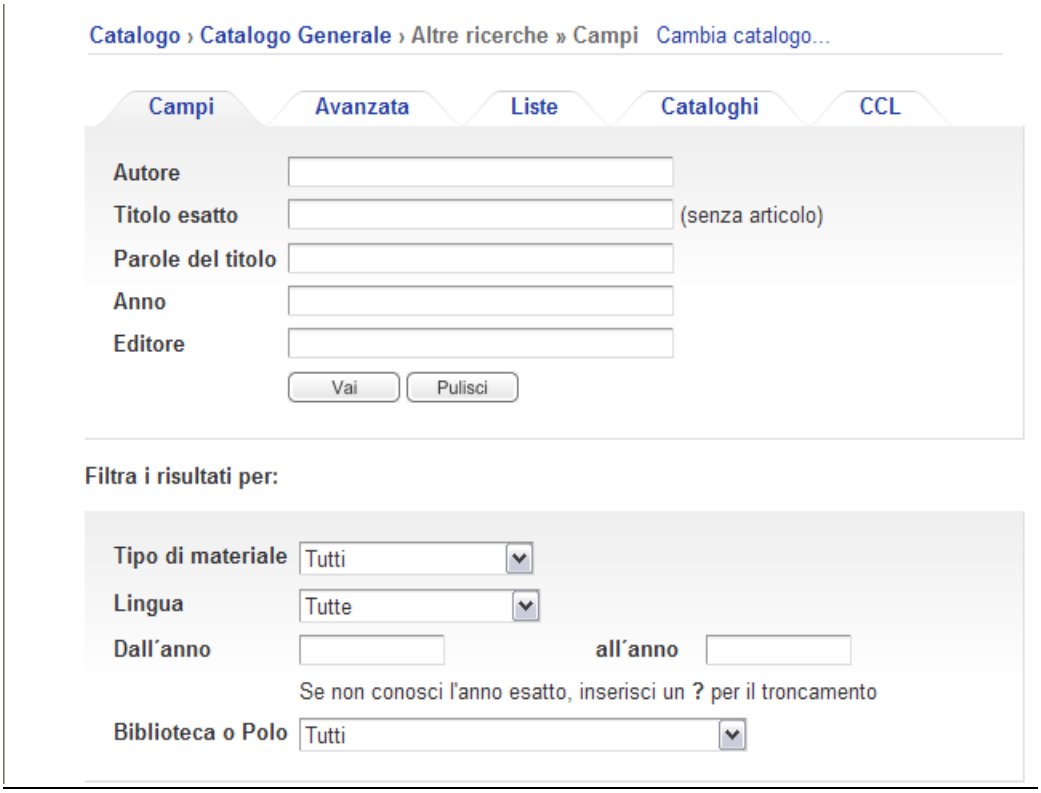


Fig 3.3: Schermata inserimento dati per altre ricerche[Cat].

Anche in questo caso è possibile filtrare i risultati per tipo di materiale, lingua di pubblicazione, anno di pubblicazione e biblioteca o polo di appartenenza.

3.2.2 Avanzata

Questa linguetta apre un nuovo form di inserimento dati che offre, rispetto alla Ricerca semplice o alla Ricerca per Campi, una gamma più ampia di opzioni di ricerca. Si possono utilizzare i menu a tendina per specificare i campi da utilizzare e definire le modalità di ricerca. Nel caso di utilizzo di più campi, la relazione tra questi è gestita attraverso l'operatore logico AND. Per ricercare parole che nel testo sono vicine ad altre, bisogna impostare Sí per l'opzione *Parole adiacenti* [Ca_Gu]. Per visualizzare la lista dei record, bisogna eseguire un click sulle cifre della colonna

di destra che, una volta premuto il tasto Vai, riportano i numeri dei risultati totali e parziali a lato di ogni chiave di ricerca. Anche in questo caso è possibile filtrare i risultati tramite le opzioni già viste negl'altri casi.

Fig 3.4: Form di inserimento dati per la Ricerca Avanzata[Cat].

3.2.3 Liste

Questa linguetta offre un form più ridotto con una casella di testo ed un menù a tendina. La casella di testo serve per la stringa di ricerca, mentre il menù serve per selezionare l'indice da consultare. La ricerca per liste permette di consultare il catalogo scorrendo elenchi alfabetici o numerici, come qunado si consulta un dizionario. La ricerca, in caso di risultato positivo, produrrà una lista alfabetico/numerica in cui saranno elencate tutte le voci che rispondono o si avvicinano alla richiesta. La stringa di ricerca apparirà all'inizio di una lista.

Per esempio, se inserisco la lettera *a* verrà visualizzata una lista alfabetica che inizia per *a*. Se inserisco la parola *grande* verrà visualizzata una lista che inizia per *grande* [Ca_Gu].

Fig 3.5: Form dati per la ricerca nelle Liste[Cat].

3.2.4 Cataloghi

Questa linguetta propone l'elenco dei cataloghi possibili in modo che l'utente possa scegliere quale consultare, quelli di ateneo, quelli delle singole biblioteche convenzionate, quelli riguardanti i vari formati. Utilizzando la ricerca per cataloghi si possono limitare le ricerche ad un solo catalogo. Si accede a Cataloghi sia cliccando sulla linguetta relativa a questo tipo di ricerca in Altre ricerche, sia cliccando sulla voce Cambia catalogo presente nelle varie funzionalità di ricerca. Una volta cambiato il catalogo di ricerca per tornare al Catalogo generale bisogna selezionarlo nella pagina dei "Cataloghi" [Ca_Gu].

3.2.5 CCL – Common Command Language

Questa linguetta apre un form in cui è possibile inserire una stringa di comandi CCL. Si può utilizzare il Common Command Language (CCL) per ricercare parole e intestazioni in indici diversi impostando un'unica stringa di ricerca. Bisogna specificare il codice degli indici da ricercare. Questi codici sono effettivi solamente nella maschera di ricerca CCL.

Esempi:

WRD=test cercherà la parola "test" in qualsiasi campo.

Si possono combinare ricerche o filtri diversi nella stringa di ricerca: *WRD=test AND WLN=ita* troverà i record di lingua italiana contenenti la parola "test" in qualsiasi campo. Con l'uso delle parentesi si possono combinare diverse richieste per creare una ricerca complessa. L'asterisco può essere utilizzato per troncare le parole:

((WAU=carlyle OR ruskin OR hegel) AND (WTI=cultur)) NOT (WSU=art*)* rintraccerà tutte le opere scritte da autori il cui nome è Carlyle oppure Ruskin oppure Hegel, il cui titolo contiene la parola *cultur*, per esempio, *cultura*, *culture*, *cultural*, e così via, ma la cui stringa di soggetto non contiene la parola *art*, per esempio *arte*, *art*, *artist*, *artistic*, ecc [Ca_Gu].

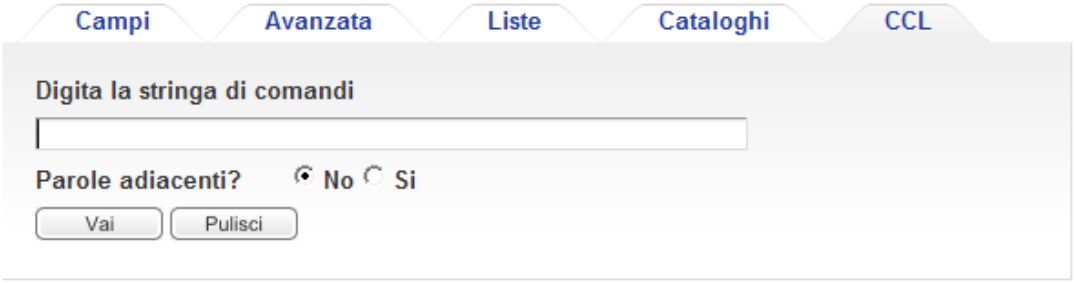


Fig 3.6: Form per l'inserimento di comandi CCL[Cat].

Anche in questo caso i risultati possono essere filtrati con le opzioni già viste. Nella guida di supporto al sito si trovano tutte le abbreviazioni in uso per i comandi CCL.

3.3 Le ultime 3 voci

Le ultime tre voci del menù principale iniziale sono utili per rielaborare e controllare le operazioni di ricerca svolte in passato tramite le varie opportunità viste prima.

3.3.1 Elabora ricerca

Questa pagina offre la possibilità di lavorare con i risultati dell'ultima ricerca effettuata. La pagina che viene offerta tramite questo link è la stessa che compare con i risultati di qualsiasi ricerca. I record trovati vengono visualizzati nella finestra della lista dei risultati. La lista dei risultati mostra quanti record soddisfano la ricerca ed ogni record nella lista è numerato progressivamente.La finestra dei risultati contiene molte informazioni relative a ciascun record:

- autore;
- formato;
- titolo;
- anno di pubblicazione;
- biblioteca di appartenenza / copie presenti;

- legame esterno;

La chiave di ricerca viene mostrata nella parte in alto a sinistra della schermata, sotto al menu principale. Si può scegliere il formato di visualizzazione e di ordinamento dei record.

Sono a disposizione due opzioni per spostarsi alla pagina del record desiderato: fare un click sul titolo dell'opera, o farlo sul numero di ordinamento dato dal sistema al record**.**

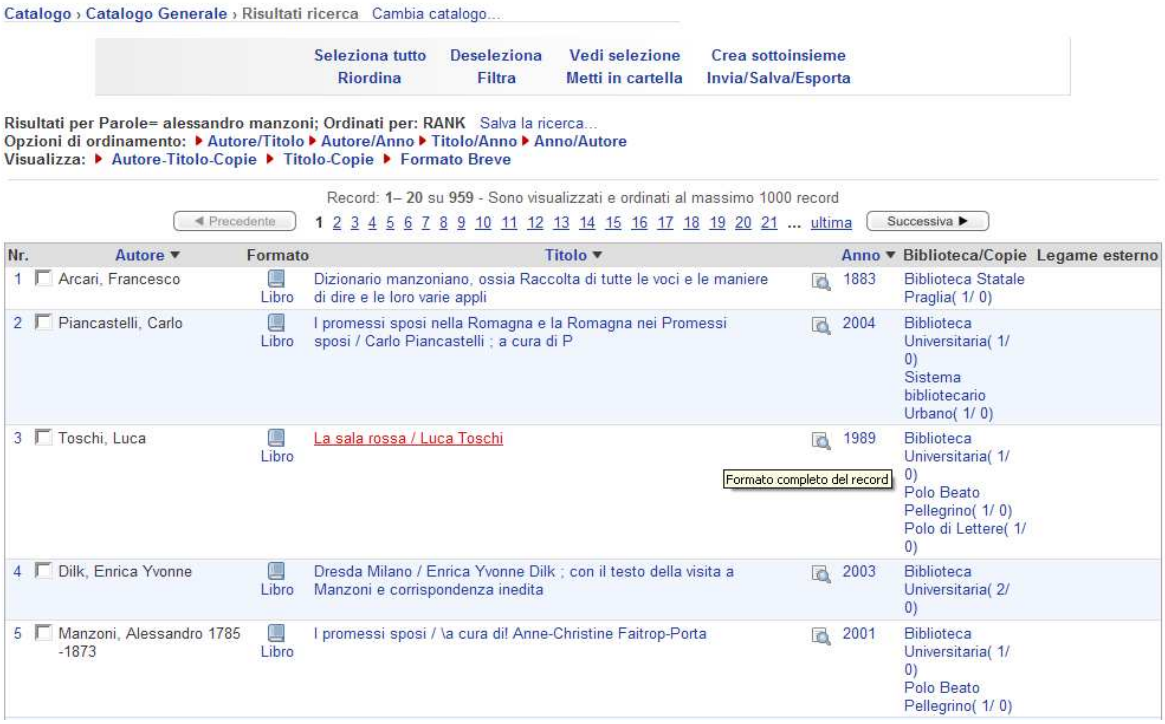


Fig 3.7: Una pagina dei risultati con menù di elaborazione dati[Cat].

In alto al centro della pagina sono disponibili diverse opzioni, descritte di seguito, che permettono di elaborare i risultati della ricerca:

- Seleziona tutto

  Questa funzione permette di selezionare automaticamente tutti i risultati della lista (comunque non più di 50 record).

- Deseleziona

  Con Deseleziona si possono deselezionare automaticamente tutti i risultati della lista che erano stati selezionati.

- Vedi selezione

  Questa funzione permette di visualizzare i record selezionati in una finestra dalla quale poi si può procedere alla loro elaborazione.

- Crea sottoinsieme

  La creazione di un sottoinsieme permette di estrarre un gruppo di record dalla lista dei risultati.Una volta selezionati i record che interessano, la lista dei risultati visualizzerà solo il sottoinsieme dei record selezionati. Il sottoinsieme potrà essere trattato come qualsiasi altra lista di risultati: i record possono essere selezionati, aggiunti ad una lista, inviati per e-mail, o trasformati in un altro sottoinsieme. Il sottoinsieme rimane nelle Ricerche eseguite sino al termine della sessione di lavoro. Dalla pagina Ricerche eseguite i sottoinsiemi possono essere combinati con altre ricerche. L'unico modo di distinguere un sottoinsieme da una lista di record completa è attraverso il numero di record che contiene poiché la stringa di ricerca non cambia [Ca_Gu].

- Riordina

  Si può modificare l'ordinamento dei record utilizzando le opzioni di ordinamento predefinite nella Lista breve oppure cliccando su un'intestazione di colonna (per esempio se si clicca sull'intestazione Autore, i record vengono ordinati per cognome dell'autore). Oltre a questi metodi si può cliccare sulla funzione Riordina per riordinare i record aggiungendo un'ulteriore parola a quelle utilizzate nella ricerca. La lista dei risultati sarà reimpostata in base ad una formula che prende in considerazione il numero di volte che la nuova parola compare in ciascun record, e il peso attribuitole (per esempio, se la parola è contenuta nel titolo, le può essere riconosciuto un peso maggiore) [Ca_Gu].

- Filtra

  La funzione Filtra permette di ampliare o restringere la ricerca. Ci sono tre possibili filtri di ricerca:

  Parole: permette di raffinare i risultati della ricerca aggiungendo altre parole e combinandole con la ricerca iniziale tramite gli operatori booleani. Se si

vuole restringere una ricerca, bisogna usare AND o NOT, altrimenti per ampliarla scegliere OR;

Intervallo di valori: se si seleziona il campo Titolo e si scrive in 'Testo da': "C" e in 'Testo a': "D" si selezionerà una lista che va dai titoli che iniziano con C fino ai titoli che iniziano con D;

Data di creazione: è la data di inserimento del record bibliografico nel Catalogo (non la data di pubblicazione)[Ca_Gu].

- Metti in cartella

  C'è la possibilità di salvare i record selezionati in una cartella temporanea. Si possono aggiungere record alla cartella partendo dalla visualizzazione completa di un record o dalla lista dei risultati. Questa opzione è legata ai permessi definiti per il profilo utente in uso e alla configurazione prevista dalla biblioteca. Si possono inviare i record per e-mail ad uno specifico indirizzo. Per visualizzare il contenuto della cartella, bisogna cliccare *La mia cartella* sulla barra del menu principale. Se l'utente non è autenticato la cartella temporanea si svuota in automatico quando termina la sessione.

- Invia / Salva / Esporta

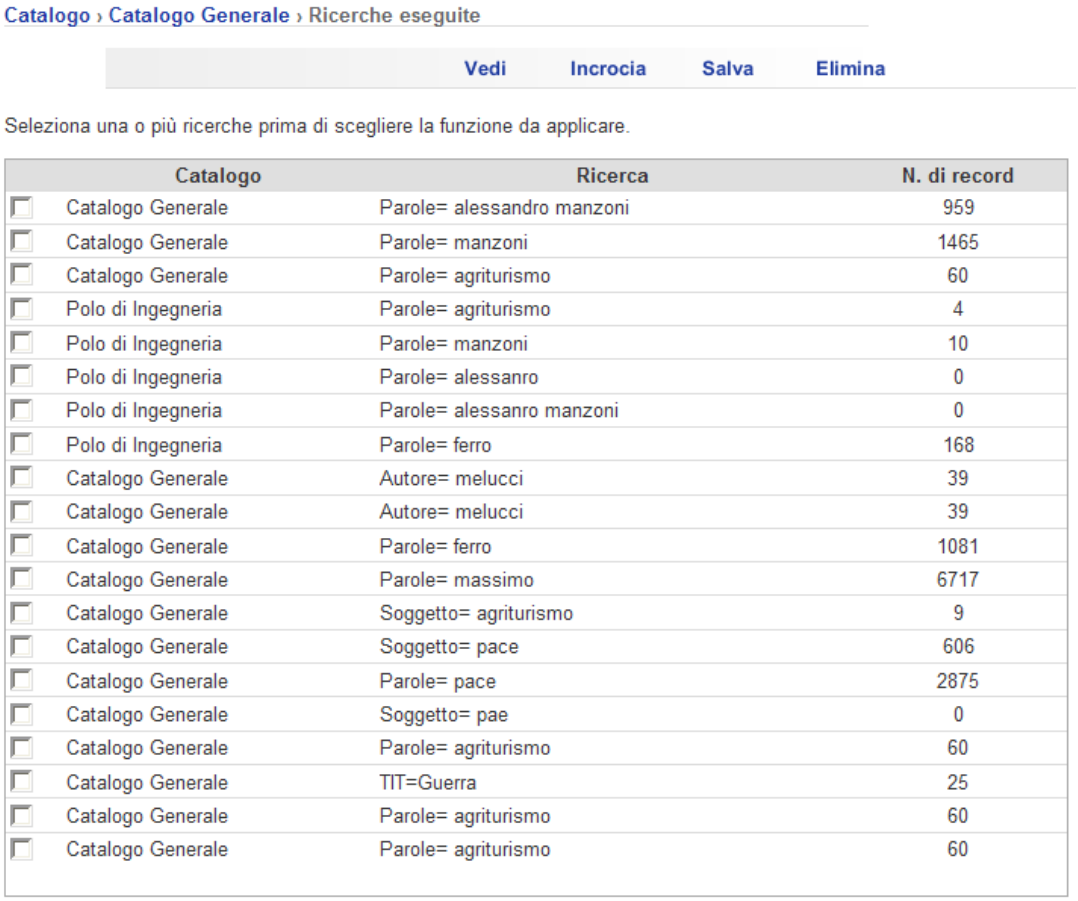
  Se sono stati selezionati dei record, si può salvarli sul proprio computer o inviarli tramite e-mail. La procedura per inviare il record o la lista dei record tramite e-mail prevede di scegliere il formato in cui formattare i record e compilare il campo per l'indirizzo e-mail. Per salvare, invece, i record nel proprio computer bisogna scegliere un formato per i record e lasciare il campo indirizzo e-mail vuoto; salvare la pagina di testo che viene presentata dal browser ottenendo così un file di testo semplice, con estensione .txt. Alcuni formati vengono salvati come testo semplice, ma con estensione .sav possono essere aperti con qualsiasi editor di testo dopo aver cambiato l'estensione da .sav a .txt [Ca_Gu].

3.3.2 Ricerche eseguite

La pagina delle Ricerche eseguite permette di vedere i dati relativi a tutte le ricerche effettuate durante la sessione in atto; in caso di autenticazione avvenuta permette di

recuperare i dati di tutte le ricerche passate. I risultati delle ricerche eseguite possono essere utilizzati per nuove ricerche o possono essere combinatie tra loro in una ricerca nuova. Per rivedere una ricerca, è necessario selezionarla, e cliccare su *Vedi*, per eliminarla cliccare su *Elimina*, per combinare insieme due o più ricerche, cliccare su *Incrocia*. *Incrocia* combina tra loro due o più ricerche eseguite.

Per esempio, se c'è una ricerca relativa ad *Alessandro* ed una a *Manzoni*, si possono combinare insieme le ricerche eseguite scegliendo un operatore logico predefinito [Ca_Gu].



Catalogo	Ricerca	N. di record
Catalogo Generale	Parole= alessandro manzoni	959
Catalogo Generale	Parole= manzoni	1465
Catalogo Generale	Parole= agriturismo	60
Polo di Ingegneria	Parole= agriturismo	4
Polo di Ingegneria	Parole= manzoni	10
Polo di Ingegneria	Parole= alessanro	0
Polo di Ingegneria	Parole= alessanro manzoni	0
Polo di Ingegneria	Parole= ferro	168
Catalogo Generale	Autore= melucci	39
Catalogo Generale	Autore= melucci	39
Catalogo Generale	Parole= ferro	1081
Catalogo Generale	Parole= massimo	6717
Catalogo Generale	Soggetto= agriturismo	9
Catalogo Generale	Soggetto= pace	606
Catalogo Generale	Parole= pace	2875
Catalogo Generale	Soggetto= pae	0
Catalogo Generale	Parole= agriturismo	60
Catalogo Generale	TIT=Guerra	25
Catalogo Generale	Parole= agriturismo	60
Catalogo Generale	Parole= agriturismo	60

Fig 3.8: Un esempio della pagina Ricerche eseguite[Cat].

3.3.3 La mia cartella

La pagina relativa a La mia cartella permette di gestire i record salvati da un utente nelle sue cartelle. I record possono essere salvati e organizzati in cartelle permanenti solo dagli utenti autenticati. Per gli utenti non autenticati sono disponibili cartelle temporanee (basket) che vengono cancellate al termine della sessione. Per salvare un

record bisogna selezionarlo nell’elenco e poi cliccare *Metti in cartella* nel menu dei comandi.. Nella funzione La mia cartella si possono ordinare i record per data e per nome di cartella. Da questa funzione si possono inviare i record tramite e-mail oppure cancellarli.

4 Un problema e la sua soluzione: il Topic drift

In questo capitolo si vuole esporre le possibili soluzioni al problema del "*topic drift*", a volte detto anche "*query drift*" [EQE]. Un utente, quando usa un motore di ricerca, tenta di esprimere la sua esigenza informativa tramite una query, che nella maggior parte dei casi non supera le due parole [SEn]; una query di questo tipo può presentare al suo interno termini che hanno molteplici significati, per cui nei risultati è facile incontrare elementi che fanno parte di contesti completamente differenti l'uno dall'altro. Un altro caso in cui si verifica questo tipo di problema è quando si ha a che fare con casi di omonimia per cui la stessa query vale per individui diversi che possono essere più o meno di interesse per l'utente perché appartenenti a contesti diversi. Un modo per risolvere questo tipo di situazioni è quello di cercare di arricchire la query con altri termini in modo da definire in maniera più chiara il contesto di interesse dell'utente. Si parla in questi casi di "*query expansion*", cioè un processo guidato dal sistema che aiuta l'utente ad esprimere in maniera più precisa la sua esigenza informativa; questo può essere totalmente automatico o prevedere l'intervento dell'utente stesso [SEn]. Per espandere la query si possono usare diversi schemi di retroazione che possono coinvolgere l'utente in maniera più o meno pesante. Si parla di "*relevance feedback*".

4.1 Query expansion

Quando l'utente si trova ad affrontare il problema del *"topic drift"*, normalmente, prova a risolvere il problema da solo cercando di migliorare la query inserita, aggiungendo nuovi termini, oppure usandone di più precisi. Esistono in realtà dei metodi di miglioramento della query (*query expansion)* che fanno in modo che sia il sistema a fare il lavoro; queste metodologie possono essere totalmente automatiche oppure prevedere l'intervento dell'utente nell'elaborazione. I metodi per affrontare questo problema si dividono principalmente in due categorie: i metodi *globali* e i metodi *locali* [IIR]. I metodi *globali* sono tecniche per l'espansione o la riformulazione della query che non prendono in considerazione i risultati ottenuti con la prima query, ma prevedono l'utilizzo di strutture dati esterne (*thesaurus)* per trovare termini sinonimi o in qualche modo collegati ai termini della query originale. I metodi *locali,* invece, cercano di migliorare la query di partenza in base ai documenti che sembrano soddisfare inizialmente tale query. Questi metodi, più utili nel nostro contesto, sono:

- Relevance feedback;
- Pseudo relevance feedback;
- Indirect relevance feedback;

4.1.1 Relevance feedback

L'idea che sta alla base del concetto di *relevance feedback* è di coinvolgere l'utente nel processo di reperimento dell'informazione in modo tale da migliorare i risultati della ricerca. In quetso caso è previsto che l'utente fornisca dei giudizi di rilevanza per quanto riguarda i documenti inizialmente reperiti, in seguito, alcuni algoritmi utilizzano questi giudizi per costruire una query migliore. I passi principali del processo sono:

- l'utente inserisce una query iniziale (semplice e corta nella maggior parte dei casi);
- il sistema reperisce un insieme di risultati;
- l'utente seleziona alcuni dei risultati come rilevanti;

- il sistema cerca di rappresentare in maniera più chiara l'esigenza informativa dell'utente, basandosi sul feedback ricevuto;
- il sistema fornisce un nuovo insieme di risultati [IIR].

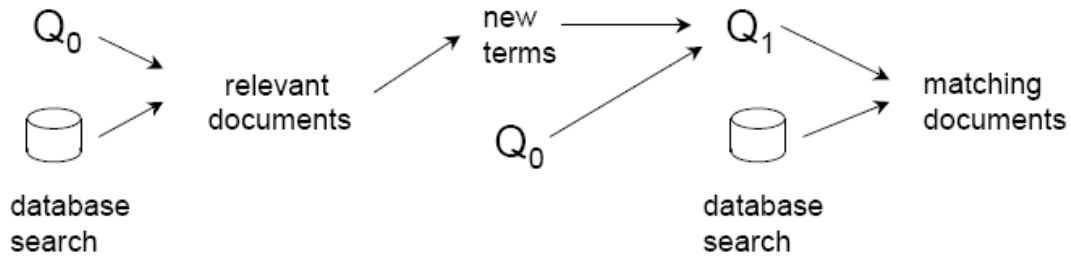


Fig: 4.1: Le fasi del processo di relevance feedback e query expansion [07Fe].

Il processo può ripetersi più volte per rifinire in modo sempre migliore la query iniziale. Per un utente può essere difficile formulare da subito la query giusta non conoscendo bene la collezione di partenza; è invece, più facile, procedere gradualmente ponendo solo dei giudizi sui docomenti reperiti e lasciando fare al sistema.

Costruire una query ottimale è difficile, se non impossibile; una query di questo tipo dovrebbe reperire tutti i documenti rilevanti ponendoli, in classifica, prima di quelli non rilevanti. Esiste un algoritmo classico per il relevance feedback che usa i giudizi di rilevanza per tentare di costruire un query quasi ottimale nel caso in cui si utilizzi il modello vettoriale: l'*algoritmo di Rocchio* [IIR].

Algoritmo di Rocchio

Rocchio propose una formula per migliorare la query iniziale grazie al contributo dei documenti giudicati rilevanti e di quelli giudicati non rilevanti:

$$Q' = aQ + b\ sum(R) - c\ sum(S)$$

dove:

Q : è la query iniziale;

R : è l'insieme dei documenti giudicati rilevanti;

S : è l'insieme dei documenti giudicati non rilevanti;

a, b, c : sono costanti (pesi);

Q' : è la nuova query.

La nuova query così ottenuta dovrebbe reperire i documenti rilevanti con un posizionamento ancora migliore. In molti casi di utilizzo della formula di Rocchio non vengono presi in considerazione i documenti non rilevanti, per cui il valore della costante c è 0. I valori tipici di a e b sono rispettivamente 1 e 8 [EQE].

Esistono altri due meccanismi di relevance feedback molto importanti sempre per il modello vettoriale. Questi due metodi sono entrambi simili alla formula di Rocchio e sono stati sviluppati da Ide.

Ide dec-hi e Ide regular

Entrambi i metodi prendono il nome dall'autore che li ha inventati. Il primo metodo, *Ide dec-hi* usa tutto l'insieme dei documenti giudicati rilevanti, come esempio positivo, e solo il primo documento dell'insieme dei non rilevanti come feedback negativo:

$$Q' = Q + \text{sum}(R) - \mathbf{s}$$

con **s** : primo documento giudicato non relevante.

Ide regular è differente solo nel considerare come esempio negativo tutto l'insieme dei documenti non rilevanti:

$$Q' = Q + \text{sum}(R) - \text{sum}(S)$$

Rispetto alla formula di Rocchio, in entrambi i casi, non ci sono castanti che fungono da pesi [EQE].

Quando, invece si ha a che fare con il modello probabilistico, si può procedere costruendo un classificatore tramite un modello probabilistico Naive Bayes [IIR].

Relevance feedback probabilistico

Usando un classificatore, nel modello probabilistico, si può cercare di usare i giudizi di rilevanza espressi dall'utente. Se si considera un indicatore booleano *R* che esprime la rilevanza di un documento si possono definire le probabilità che un termine *t* appaia in un documento rilevante o in un documento non rilevante come:

$$P(t=1 \mid R=1) = |VRt| / |VR|$$
$$P(t=1 \mid R=0) = (dft - |VRt|) / (N - |VR|)$$

dove: *N* : è il numero totale di documenti;

dft : è il numero di documenti che contiene il termine *t*;

VR : è l'insieme dei documenti rilevanti;

VRt : è l'insieme dei documenti rilevanti che contengono il termine *t* [IIR].

Anche nel caso in cui l'insieme dei documenti giudicati rilevanti sia piccolo queste probabilità possono essere una buona base per classificare i termini rilevanti eventualmente utili per espandere una query.

Come si può notare dalle formule appena viste l'approccio di relevance feedback richiede che i documenti rilevanti siano in qualche modo simili tra di loro, soprattutto per quanto riguarda il modello vettoriale.

Il relevance feedback può avere dei problemi di natura pratica in quanto query molto lunghe, generate dal processo, possono risultare inefficienti per alcuni sistemi di reperimento dell'informazione; per questo può risultare utile limitare il numero di termini da aggiungere nell'espansione della query. Il relevance feedback prevede un intervento manuale dell'utente; in alcuni casi l'utente può fare confusione nella scelta dei documenti rilevanti allontanandosi così dall'obiettivo della sua ricerca, per questo motivo molti sistemi adottano un meccanismo di relevance feedback automatico.

4.1.2 Pseudo relevance feedback & Indirect relevance feedback

Questi meccanismi di relevance feedback prevedono di rendere automatica la parte di selezione dei documenti rilevanti, in questo modo l'utente non ha più un ruolo determinante nel processo, a parte la definizione della query iniziale. Il proposito di questi metodi è lo stesso del relevance feedback semplice, cioè migliorare la query di partenza per cercare di reperire la maggior parte dei documenti rilevanti per l'esigenza informativa dell'utente.

Con il *pseudo relevance feedback* si esegue una prima ricerca sulla query, poi si assume che i primi *k* documenti recuperati siano rilevanti e si procede con una delle formule viste [IIR]; in questo caso è il sistema che decide quali sono i documenti rilevanti e quali no basandosi sul fatto che normalmente i documenti più rilevanti, per la query inserita, si trovano nelle prime posizioni del ranking dei risultati. Il problema più importante, con questo tipo di feedback, consiste nel considerare erroneamente rilevanti documenti che in realtà non lo sono, solo perché si trovano nelle prime *k* posizioni.

Con l'*indirect relevance feedback* si usano delle risorse indirette, legate ai comportamenti degli utenti, per ottenere un feedback [IIR]. Questo metodo è meno affidabile del relevance feedback esplicito, ma è più utile del pseudo relevance feedback che non contiene informazioni sui giudizi dell'utente. Anche se spesso gli utenti non gradiscono compiere operazioni aggiuntive per esprimere dei giudizi di rilevanza, è facile collezionare feedback impliciti dai dati relativi alle loro sessioni web. Un modo per raccogliere i dati utili per il feedback è quello di assumere che un click su un link sia indicativo del fatto che la pagina di destinazione è rilevante per la query inserita. Una volta raccolti questi dati si potranno definire le pagine rilevanti per una query grazie ai comportamenti passati degli utenti e procedere così all'espansione della query lavorando con questi documenti.

Il relevance feedback, con uno qualsiasi degli schemi appena visti, permette di seguire, in maniera appropriata, i cambiamenti dell'esigenza informativa dell'utente dovuti al variare degli interessi dell'utente stesso durante la navigazione.

4.2 Term selection: la pesatura

Dopo aver applicato uno degli schemi appena visti di relevance feedback, per procedere nell'espandere la query bisogna estrarre i termini dai documenti considerati rilevanti e pesarli in modo da poter ricavare un indice ordinato. La pesatura deve garantire che i termini nelle prime posizioni dell'indice siano quelli più utili per espandere la query in modo da avvicinarsi agli interessi dell'utente. Nel campo del reperimento dell'informazione sono stati proposti diversi algoritmi che cercano di quantificare l'utilità di un termine al contesto di interesse. Esistono delle

relazioni tra la frequenza di un termine , la sua utilità nel contesto, e la sue efficienza nel reperire documenti rilevanti:

- termini molto frequenti non sono utili;
- termini di media frequenza sono abbastanza utili;
- termini poco frequenti sembrano essere utili, ma non quanto quelli di media frequenza;
- termini rari sono molto utili in quanto sono garanzia di rilevanza. Nella maggior parte dei casi, però, non permettono di reperire molti documenti in realtà rilevanti [Eft93].

Da queste relazioni si capisce che un buon algoritmo dovrebbe prendere i termini di media frequenza e porli nelle prime posizioni dell'indice da lui creato. Esistono diversi algoritmi per la pesatura, ma per quanto riguarda la selezione di termini per la query expansion , si fa riferimento principalmente a questi 6:

4.2.1 Algoritmo f4

La teoria della pesatura dei termini pone le sue basi sui giudizi di rilevanza. Tale teoria si basa su due assunzioni:

- assunzione d'indipendenza : la distribuzione dei termini nei documenti rilevanti è indipendente, così come nei documenti non rilevanti;
- assunzione d'ordinamento: la probabilità di rilevanza di un documento è basata sia sulla presenza dei termini della query, nel documento stesso, che sulla loro assenza.

La formula base di questa teoria, conosciuta anche come *BIM (Binary Independence retrival Model)* [Eft95] è:

$$w = \log \frac{p*(1-q)}{q*(1-p)}$$

con p : probabilità che il termine sia presente in un documento rilevante,

q : probabilità che il termine sia presente in un documento non rilevante.

Applicare questa formula richiede alcune conoscenze relative alle presenze dei vari termini nei documenti rilevanti e non. Le probabilità p e q possono essere stimate grazie ai dati di relevance feedback, anche considerando una sola parte dei

documenti rilevanti. Invece di usare le probabilità è più utile usare le frequenze che sono più facili da trovare. Per evitare di avere pesi infiniti si usa un piccolo incremento di correzione in ciascuna componente della formula. La formula che ne risulta viene detta *f4 :*

$$\mathbf{f4 = \log \frac{(r + 0.5) * (N - n - R + r + 0.5)}{(n - r + r + 0.5) * (R - r + 0.5)}}$$

dove *r* : è il numero di documenti rilevanti, tra gli R, che contengono il termine considerato;

R : è il campione di documenti rilevanti definiti dal relevance feedback;

n : è il numero di documenti, in totale, che contengono il termine;

N : è il numero di documenti della collezione [Eft 95].

4.2.2 Algoritmo di Porter

Porter e Galpin nel 1988 usarono una nuova formula:

$$\mathbf{porter = \frac{r}{R} - \frac{n}{N}}$$

dove *r, n, R,* ed *N* sono definiti come nell'algoritmo f4. Non si trovano spiegazioni di come arrivarono a questa formula [Eft 95]. Si può comunque osservare che la frazione *r / R* non diverrà mai 0, in quanto ci sarà sempre almeno un documento considerato rilevante che contiene il termine; e assumerà valore massimo 1 nel caso in cui *r = R*, cioè tutti i documenti giudicati rilevanti contengano il termine. La formula di Porter sembra dare molta più importanza ai termini che compaiono molte volte nei documenti considerati rilevanti; questo perché la componente principale della formula è, per l'appunto, la frazione *r / R* .

4.2.3 Algoritmo emim

L'algoritmo *emim* (Expected Mutual Information Measure) è uno schema di pesatura che sfrutta le informazioni di rilevanza, ma assume che i termini indice non siano distribuiti indipendentemente l'uno dall'altro:

$$emim = E_{iq} = \sum_{t_i, w_q} \Delta_{iq} P(t_i, w_q) \log \frac{P(t_i, w_q)}{P(t_i) P(w_q)}$$

dove t_i : indica la presenza (1) o assenza (0) del termine;

w_q : indica se il documento è rilevante (1) o non rilevante (0);

Δ_{iq} : indica il valore del termine come discriminatore rilevante e vale 1 nel caso in cui $t_i = w_q$, 0 se $t_i \neq w_q$ [Eft 95].

Nel caso in cui le probabilità congiunte siano tutte unitarie la formula si riduce a quella dell'algoritmo f4. Con le stesse definizioni di *r*, *n*, *R* ed *N* il peso di un termine, con la fromula di questo algoritmo diventa:

$$E_{iq} = p_{11}i_{11} - p_{12}i_{12} - p_{21}i_{21} + p_{22}i_{22}$$

$$= \log \frac{rN}{Rn} \cdot r$$

$$- \log \frac{(n-r)N}{(N-R)n} \cdot (n-r)$$

$$- \log \frac{(R-r)N}{(N-n)R} \cdot (R-r)$$

$$+ \log \frac{(N-n-R+r)N}{(N-n)(N-R)} \cdot (N-n-R+r)$$

4.2.4 Algoritmo wpq

Nel 1990 fu sviluppato da Robertson un nuovo algoritmo per la pesatura dei termini dati dal relevance feedback [Eft93]. Oltre all'assunzione di indipendenza dei termini all'interno dei documenti, nella query expansion deve essere fatta un'altra assunzione: l'indipendenza statistica tra il termine usato per la query expansion e i termini utilizzati nelle ricerche precedenti. L'insieme dei documenti rilevanti nella prima ricerca può essere diviso in due sottoinsiemi: da una parte i documenti rilevanti che contengono il termine considerato per l'espansione, e dall'altra i documenti rilevanti che non contengono tale termine. Questi due sottoinsiemi vengono considerati allo stesso modo, per quanto riguarda la ricerca iniziale, grazie all'assunzione fatta sull'indipendenza statistica. L'inclusione del termine *t* nella query, con peso w_t porterà un incremento in efficienza del reperimento pari a:

$$wpq = w_t(p_t - q_t)$$

dove w_t : è il peso assegnato al termine con uno specifico schema di pesatura, in questo caso f4;

p_t : è la probabilità che il termine *t* compaia in un documento rilevante;

q_t : è la probabilità che il termine *t* compaia in un documento non rilevante [Eft95].

Questo significa che, a prescindere dalla funzione usata per calcolare w_t, la regola per decidere l'inclusione del termine nella query expansion deve essere basata sui valori di *wpq* invece che del solo w_t. Ora sostituendo la funzione di pesatura e le probabilità con i soliti valori *r, n, R,*ed *N*:

$$wpq = \log \frac{(r+0.5)(N-n-R+r+0.5)}{(n-r+0.5)(R-r+0.5)} \cdot \left(\frac{r}{R} - \frac{n-r}{N-R} \right)$$

La componente w_t della formula assegna grande importanza ai termini non frequenti; in realtà un modello che definisce i termini per la query expansion dovrebbe preferire una soluzione intermedia tra i termini molto frequenti e quelli rari. Nella formula di *wpq* questo è garantito dalla componente $p_t - q_t$. Questa componente è influenzata dalla frequenza di un termine nei documenti rilevanti, così come dalla frequenza dello stesso termine nell'intera collezione. Il prodotto tra le due componenti produce un effetto che sembra assomigliare a quello desiderato per un modello di query expansion.

4.2.5 Algoritmo r-lohi

L'algoritmo r-lohi nasce nel 1993, con Efthimiadis [Eft93], come risultato dell'osservazione del comportamento degli altri algoritmi di pesatura per la query expansion.

L'algoritmo prevede di:

- classificare i termini in base alla loro frequenza *r*, per esempio in ordine decrescente nei documenti rilevanti, e
- risolvere i conflitti, di pari frequenza *r*, in accordo con la loro frequenza *n*, prima i termini di frequenza minore e poi a seguire con l'aumentare della frequenza.

L’abbreviazione *r-lohi* significa priprio questo *r-low-to-high*. L’algoritmo è nato con l’ipotesi di seguire un modello di prestazione simile a quello di *wpq*; in realtà l’algoritmo si comporta in maniera totalmente diversa nel caso in cui l’insieme dei documenti rilevanti sia molto grande. In caso di pari frequenza *r* vengono favoriti i termini meno frequenti nella collezione.

4.2.6 Algoritmo r-hilo sort

L’algoritmo *r-hilo sort* è una variante dell’algoritmo *r-lohi* in cui si deve:

- classificare i termini in base alla loro frequenza *r*, in ordine decrescente, nei documenti rilevanti, e
- risolvere i conflitti, di pari frequenza *r*, in accordo con la loro frequenza *n*, prima i termini di frequenza maggiore e poi a seguire con il diminuire della frequenza [Eft 93].

L’abbreviazione *r-hilo* sta proprio a significare *r-high-to-low*. Questo algoritmo classifica i termini in conflitto, per la frequenza *r*, esattamente al contrario rispetto a *r-lohi*, favorendo i termini più frequenti.

5 L’Applicazione

In questo capitolo viene presentata l’applicazione realizzata: un’ estensione per Mozilla Firefox che lavora su Google e usa la query inserita dall’utente, nel motore di ricerca, per effettuare, la stessa ricerca, nel sito del Catalogo dell’OPAC dell’Università di Padova. I risultati dell’OPAC vengono analizzati e presentati a fianco dei risultati di Google, al posto dei link sponsorizzati, in modo da potersi collegare direttamente agli elementi che più interessano l’utente.

La seconda fase dell’estensione nasce per risolvere un problema che si è visto comparire nelle varie ricerche di Google: il “*topic drift*” [EQE]. Per risolvere questo tipo di situazioni si è deciso di cercare di arricchire la query con altri termini in modo da definire in maniera più chiara il contesto di interesse dell’utente tramite un processo di ”*query expansion*”. Per fare tutto ciò si è partiti dall’assunzione che i risultati forniti dall’OPAC sono di ottima qualità e che per l’utente possono essere, nella maggior parte dei casi, validi per rappresentare il suo contesto di interesse. In questo modo si è cercato di utilizzare tali risultati per espandere la query iniziale di Google e migliorare così i risultati ottenuti dal motore di ricerca.

Per espandere la query si è scelto di coinvolgere l’utente nella definizione del contesto di ricerca, per cui si usa il “*relevance feedback*”, cioè una retroazione esplicita. Il progetto prevede che a fianco di ogni risultato fornito dall’OPAC ci sia un checkbox che l’utente può spuntare se l’elemento è di suo interesse; sotto i risultati c’è un tasto che permette di espandere la query iniziale con le informazioni

scelte dall'utente. In questo caso, quindi, c'è una retroazione in cui l'utente, tramite i checkbox esprime dei giudizi di rilevanza sui risultati dell'OPAC e permette di estrarre, da questi, nuovi termini da aggiungere alla query iniziale in Google per migliorarne i risultati di ricerca. Ogni elemento dell'OPAC esprime il suo contesto tramite due informazioni che sono il titolo e gli autori; il progetto prevede che queste informazioni vengano analizzate tramite un processo di indicizzazione tipico dell'*information retreival* [ReIn]: il primo passo consiste nell'eliminazione delle stop word, il secondo nello stemming e l'ultimo nella pesatura.

Per eseguire l'eliminazione delle stop word si è deciso di usare una stoplist di termini comuni presa direttamente dal motore di ricerca Google e memorizzata in un file di testo (in appendice). Tutte le parole presenti in titolo e autore, che compaiono anche nella stoplist non vengono considerate, così da poter eliminare la maggior parte degli articoli e delle preposizioni, indifferentemente dalla lingua (inglese - italiano), che non sono importanti al fine di reperire informazione utile.

Per lo stemming si è deciso di non fare alcuna operazione visto il numero limitato di parole che esprimono il contesto e quindi la scarsa probabilità di trovare radici comuni tra tali termini. Un secondo problema è rappresentato dalla lingua, infatti servirebbe uno stemmer diverso per ogni lingua, e in ogni caso non sarebbe facile riconoscere la lingua di ogni record.

Per quanto riguarda la pesatura dei termini utilizzare gli schemi tradizionali del reperimento dell'informazione non è una soluzione adeguata; infatti lo schema *TF (Term Frequency)* è banale perché avremmo un peso unitario nella maggior parte dei casi; usare invece lo schema *IDF (Inverse Document Frequency)* [ReIn] risulterebbe di scarsa efficacia in quanto la collezione di riferimento, analizzata di volta in volta, cambia di numerosità, ma soprattutto, perché questo schema, avendo collezioni molto limitate, non permette di discriminare effettivamente i termini rilevanti in base ai giudizi dell'utente. Per risolvere quindi questo problema si è ricorso agli schemi di pesatura appositi per la *"query expansion"* che si trovano in letteratura e che sfruttano le informazioni del *"relevance feedback"* , scegliendo come schema definitivo l'algoritmo *wpq* che fa uso dell'algoritmo *f4.* Questi algoritmi sono molto efficaci perché, anche prendendo in considerazione piccole collezioni [Eft93], come l'insieme dei risultati OPAC visualizzati, permettono di stimare le probabilità che un

termine compaia in un documento rilevante o non rilevante [Eft92] e di identificare in maniera corretta i termini più importanti nel contesto analizzato, in base ai giudizi espressi dall'utente.
Una volta fatte queste operazioni i termini pesati vengono raccolti in una specie di dizionario ordinato in modo da avere ai primi posti i termini con i pesi maggiori. I termini, che hanno lo stesso peso, nel dizionario, a partire dal primo, se non erano già presenti nella query iniziale, vengono aggiunti a quest'ultima e la nuova query, così formata, viene utilizzata per effettuare una nuova ricerca in Google che porterà anche ad una nuova ricerca nell'OPAC, ricominciando così il processo appena descritto. Se, invece, un termine tra quelli di maggior peso, del dizionario ordinato, era già presente nella query iniziale si passa a considerare il termine successivo senza inserirlo nella query. Ad ogni iterazione dell'intero processo di *"query expansion"*, viene aggiunto alla query iniziale, se possibile, almeno un termine di peso positivo per migliorare i risultati della ricerca in Google. In questo modo si restringe di volta in volta il contesto di interesse dell'utente.
Nel caso in cui la ricerca nel sito dell'OPAC con la query iniziale, o successivamente con la query espansa, fornisca un solo risultato non viene attuata la procedura appena descritta, per l'espansione della query, in quanto si ritiene che la query abbia identificato correttamente il contesto di interesse dell'utente e per questo non presenti risultati in cui si possano identificare contesti tra loro differenti.
Nel resto del capitolo, si procede con un'analisi più dettagliata dei vari componenti dell'estensione. Nella prima parte viene descritto il lavoro relativo alla parte grafica e strutturale dell'estensione, facendo riferimento al lavoro dell'ing. Davide Cisco [DaCi]; nella seconda parte del capitolo si spiegano le funzioni dei file JavaScript che sono il cuore dell'estensione e che eseguono tutti i passi visti in precedenza; poi nell'ultima parte vengono analizzate tutte le problematiche affrontate per realizzare l'estensione e le varie scelte progettuali.

5.1 Scheletro dell'estensione

Per realizzare la mia estensione, chiamata Catalogo_1.0, mi sono appoggiato al lavoro dell'ing. Davide Cisco che aveva realizzato un'estensione Mozilla,

SearchNote, che recuperava i risultati dell'OPAC della biblioteca di Firenze [DaCi]. Lo scheletro della mia estensione quindi, ricalca quello dell'estensione SearchNote, e il codice sorgente dei file di struttura è un adattamento di quello sviluppato da Davide Cisco [DaCi].

5.1.1 File `install.rdf` e `chrome.manifest`

Per prima cosa ho creato i file `install.rdf` e `chrome.manifest`; i due manifesti di installazione che sono indispensabili per qualsiasi estensione Mozilla si voglia creare.

```
1  <?xml version="1.0"?>
2
3  <RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4                  xmlns:em="http://www.mozilla.org/2004/em-rdf#">
5
6          <Description about="urn:mozilla:install-manifest">
7                  <em:name>Catalogo</em:name>
8                  <em:version>1.0</em:version>
9                  <em:description>Strumento di personalizzazione
10                                 per un motore di ricerca</em:description>
11                 <em:creator>Alex Villatora</em:creator>
12                 <em:id>catalogo@alexvillatora.com</em:id>
13
14                 <em:targetApplication>
15                         <!-- Firefox -->
16                         <Description>
17                                 <em:id>{ec8030f7-c20a-464f-9b0e-13a3a9e97384}</em:id>
18                                 <em:minVersion>1.5</em:minVersion>
19                                 <em:maxVersion>3.6.*</em:maxVersion>
20                         </Description>
21                 </em:targetApplication>
22
23                 <em:iconURL></em:iconURL>
24                 <em:homepageURL></em:homepageURL>
25         </Description>
26 </RDF>
```

Fig 5.1: Codice del file `install.rdf`.

Il file `install.rdf` permette di specificare a Firefox i dettagli di installazione dell'estensione. I dati da inserire per l'estensione che si sta creando sono:

- nome, versione e descrizione dell'estensione (righe 7-8-9);
- nome dell'autore (riga 11);
- identificativo dell'estensione: si usa il nome dell'estensione, seguito dal carattere @, e dal dominio principale del sito dello sviluppatore; in realtà non

serve possedere un dominio, si può inserire un qualsiasi dominio tipo *iltuonomecompleto.com* [TuEx](riga 12);

- la versione massima di Firefox che deve supportare l'estensione: è necessario aggiornarla alle ultime versioni disponibili, altrimenti l'estensione non funzionerà in tutti i browser; (riga 19);
- l'indirizzo di una pagina web in cui reperire informazioni sull'estensione, se tale pagina esiste (riga 24);

Il secondo file di installazione, `chrome.manifest`, serve per indicare dove si trovano i componenti dell'estensione.

```
1 content catalogo        jar:chrome/catalogo.jar!/content/
2 overlay chrome://browser/content/browser.xul    chrome://catalogo/content/catalogo.xul
3 locale  catalogo        en-US           jar:chrome/catalogo.jar!/locale/en-US/
4 locale  catalogo        it-IT           jar:chrome/catalogo.jar!/locale/it-IT/
5 skin    catalogo        classic/1.0 jar:chrome/catalogo.jar!/skin/
```

Fig 5.2: Codice del file `chrome.manifest`.

Nel file si trovano indicazioni riguardanti:

- la directory con i file di implementazione dell'interfaccia (riga 1);
- quali sono i componenti del browser che vengono modificati dall'estensione (riga 2),
- le lingue disponibili (righe 3-4);
- i file relativi all'aspetto grafico dell'interfaccia (riga 5).

Il passo successivo è stato quello di creare il file necessario per poter aggiungere il componente relativo all'estensione nel browser.

5.1.2 Aspetto dell'estensione

Per Catalogo è stato aggiunto un pulsante-menù vicino alla barra degli indirizzi in modo da poter attivare o disattivare l'estensione. Per fare questo è stato necessario esaminare i file XUL del browser per trovare il punto in cui inserire il pulsante. Questi file, sono scritti in linguaggio *XUL (XML-based User interface Language)*, cioè un linguaggio per interfacce basato su XML, e servono per definire la struttura dell'estensione. XUL permette di sviluppare nuovi componenti attraverso gli *overlay:* in sostanza scrivere delle direttive che dovranno andare a sovrapporsi a

quelle di un componente già esistente, senza modificare il codice di quest'ultimo. Il file creato per questo proposito si chiama `catalogo.xul`.

```
1 <?xml version="1.0"?>
2
3 <?xml-stylesheet href="chrome://global/skin/global.css" type="text/css"?>
4 <?xml-stylesheet href="chrome://catalogo/skin/catalogo.css" type="text/css"?>
5
6 <!DOCTYPE overlay SYSTEM "chrome://catalogo/locale/catalogo.dtd">
```

Fig 5.3: Prime righe del file `catalogo.xul`.

Nelle prime righe di questo file si specifica dove si trovano i file per lo stile (righe 3-4, file di tipo *css*) e quelli per la lingua (riga 6). Nella seconda parte del file si trova la definizione del punto in cui inserire il pulsante e le direttive per definire i componenti aggiuntivi dell'estensione. Per trovare il punto in cui inserire l'estensione è stato utilizzato un piccolo programma, il DOM Inspector [DOM], che permette di analizzare la struttura ad albero di ciascun documento basato su XML e recuperare gli identificativi dei vari nodi oltre alle loro proprietà. Con DOM Inspector sono stati esaminati i file XUL del browser identificando il nodo che contiene la barra degli indirizzi; di questo nodo è stato prelevato l'*id* per inserirlo nel file `catalogo.xul` come *merge point*, cioè come elemento della componente principale, in questo caso il browser, che deve essere modificato (riga 16).

```
 8 <overlay id="catalogo" xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
 9         <script type="application/x-javascript" src="chrome://catalogo/content/catalogo.js" />
10         <script type="application/x-javascript" src="chrome://catalogo/locale/language_data.js" />
11         <script type="application/x-javascript" src="chrome://catalogo/content/libraries/opac_search.js" />
12         <script type="application/x-javascript" src="chrome://catalogo/content/engines/google_processor.js" />
13
14
15         <!-- pulsante con comandi estensione -->
16         <toolbar id="nav-bar">
17                 <toolbarbutton id="catalogo-main-button" type="menu"
18                                         label="&smb-label;" tooltiptext="&smb-tooltip;"
19                                         insertafter="search-container">
20                         <menupopup>
21                                 <menuitem id="catalogo-activate" type="checkbox" checked="true"
22                                                 label="&sa-label;" tooltiptext="&sa-tooltip;"
23                                                 accesskey="&sa-accesskey;" persist="checked" />
24                         </menupopup>
25                 </toolbarbutton>
26         </toolbar>
27 </overlay>
```

Fig 5.4: Definizione dei componenti nel *merge point.*

All'interno del *merge point* sono stati definiti i componenti dell'estensione che consistono nel pulsante-menù e nella voce del menù che permette di attivare Catalogo (righe 17-25). Tra gli attributi del pulsante-menù, elemento `toolbarbutton` (riga 17), c'è da segnalare `insertafter` che permette di stabilire dove porre il pulsante indicando l'*id* dell'elemento adiacente precedente. Gli attributi `label` e `tooltiptext` (testo visualizzato quando il mouse si posiziona sul pulsante) sono specificati da entità XML che fanno riferimento al file `catalogo.dtd`. Questo file contiene i valori di riferimento degli attributi del file XUL; per ogni lingua per cui è stata predisposta l'estensione bisogna scrivere un file `catalogo.dtd` con la traduzione dei singoli valori nelle varie lingue.

```
1 <!-- pulsante principale -->
2 <!ENTITY smb-label "Catalogo">
3 <!ENTITY smb-tooltip "Catalogo - menu principale">
```

Fig 5.5: Gli attributi del pulsante nel file `catalogo.dtd` per l'italiano.

Nei file fin qui realizzati sono stati definiti solamente il tipo e la posizione dei componenti, ma non il loro aspetto grafico. Per fare ciò è necessario usare del *CSS (Cascading Style Sheet).* Il file creato è lo stesso citato nel file XUL e cioè il file `catalogo.css`. In questo file viene aggiunta un'icona al pulsante-menù in modo da renderlo riconoscibile.

```
1 #catalogo-main-button
2 {
3         list-style-image: url("chrome://catalogo/skin/mainicon.gif");
4 }
```

Fig 5.6: Codice CSS per la grafica dell'estensione.

Dopo che la struttura dell'estensione è stata definita sono stati realizzati gli algoritmi ed i file JavaScript contenenti il vero motore dell'estensione stessa. Per rendere validi i file Javascript all'interno dell'estensione è necessario dichiarare la loro presenza nel file XUL (righe 9-12 della fig. 5.4).

5.2 I file JavaScript

Le funzioni scritte nei file JavaScript dell'estensione non sono molto diverse da quelle che si scrivono abitualmente per il web. JavaScript interagisce con il browser e con la pagina caricata in esso, tramite i metodi del *DOM (Document Object Model)*

[JaSc]. Il metodo principalmente usato è `getElementById()` che, inserito come parametro l'identificativo (*id*) di un elemento da trovare, restituisce un riferimento all'elemento stesso, se questo esiste, oppure `undefined` in caso contrario. Il metodo può essere applicato a due diversi tipi di oggetto:

- se applicato a `document`, l'elemento viene ricercato tra quelli del codice XUL caricato nel browser (compresi quelli dell'estensione);
- se si cerca un elemento del file HTML caricato nel browser, per modificarne le proprietà, il metodo va applicato a `window.content.document`.

Una volta entrato in un nodo si può navigare nella struttura ad albero tramite i metodi:

- `parentNode` : per accedere al nodo padre;
- `previousSibling` : per accedere al nodo precedente dello stesso livello, avendo lo stesso padre;
- `nextSibling` : per accedere al nodo successivo dello stesso livello, avendo lo stesso padre;
- `childNodes` : per ottenere un array di riferimenti ai nodi figlio del livello inferiore.

Nell'estensione vengono creati nuovi nodi e per questo si usa il metodo `createElement()`. L'unico parametro richiesto è il tipo del nuovo elemento; questo metodo restituisce un riferimento al nuovo nodo. Il nodo così creato resta però svincolato dal resto dell'albero; per inserirlo nella struttura sarà necessario utilizzare uno dei seguenti metodi:

- `insertBefore (newNode, refNode)` : permette di inserire il nuovo nodo `newNode` prima del nodo di riferimento `refNode`;
- `appendChild (newNode)` : permette di inserire il nuovo nodo `newNode` come ultimo figlio del genitore a cui è applicato.

Attraverso DOM è anche possibile eliminare nodi (`removeChild`), sostituirli (`replaceChild`) oppure impostarne gli attributi (`setAttribute`), ottenerne i valori (`getAttribute`) od eliminarli (`removeAttribute`).

Per i nodi di elementi HTML è possibile specificare in un solo comando JavaScript il contenuto del nodo direttamente sotto forma di codice HTML; basta utilizzare la

proprietà del nodo `innerHTML`; in questo modo non serve ogni volta creare un nuovo nodo, ma basta modificarne il codice HTML [JaSc].
Il codice sorgente dei primi tre file JavaScript e' un adattamento di quello sviluppato da Davide Cisco [DaCi]. Vediamo ora gli algoritmi dell'estensione partendo dal file `catalogo.js` che permette all'estensione di compiere i primi passi.

5.2.1 File `catalogo.js`

Il file `catalogo.js` è il principale per quanto riguarda l'avvio dell'estensione.
Le funzioni contenute nel file sono, nel dattaglio:

- `Catalogo_isEnabled()` : verifica se l'estensione è abilitata;
- `Catalogo_removeWaitMessage()` : è una funzione che rimuove il messaggio di attesa visualizzato dall'estensione mentre la ricerca è in corso.

Ci sono poi altri elementi fondamentali per l'estensione:

- `Catalogo_runMeOnPageLoad` : è l'oggetto che contiene le funzioni necessarie per l'inizializzazione (`init`) e l'esecuzione (`onPageLoad`) degli script dell'estensione ogni volta che la pagina presente nel browser viene modificata;
- `Catalogo_basicSearchCompleted` : è una variabile booleana che serve per indicare se la ricerca nell'OPAC è conclusa.

La prima cosa che deve fare l'estensione è quella di istruire il browser Firefox in modo tale che al caricamento di una pagina dei risultati di Google vengano eseguiti gli script giusti. Tutto ciò è reso possibile assegnando una funzione al gestore di eventi `load` della finestra, filtrando però le pagine di nostro interesse.
Dal codice si può vedere come alla riga 5 si definisce la procedura di inizializzazione, che viene poi assegnata alla finestra del browser (`window`) alla riga 1. In questa procedura viene localizzato l'oggetto del browser relativo al contenuto delle pagine visualizzate (`appcontent`) e viene aggiunta la funzione per l'esecuzione (`onPageLoad`) al gestore di eventi `load` di quest'ultimo. Questa funzione è definita nelle righe 12-32 per svolgere i seguenti compiti:

- verifica se l'estensione è abilitata (riga 13);
- verifica se ci si trova in una pagina dei risultati di Google (riga 14);

- in caso affermativo aggiunge lo scheletro del codice HTML (riga 17) da completare poi con i risultati della ricerca (righe 19-23),
- assegna una funzione al gestore di eventi `click` del tasto submit dell'estensione che permetterà di espandere la query in Google.

```
1  window.addEventListener("load", function()
2   { Catalogo_runMeOnPageLoad.init(); },  true);
3
4  var Catalogo_runMeOnPageLoad = {
5          init: function() {
6             var appcontent = document.getElementById("appcontent"); // browser
7             if(appcontent)
8                appcontent.addEventListener("load", this.onPageLoad, true);
9
10         },
11
12         onPageLoad: function(aEvent) {
13            if (!Catalogo_isEnabled ()) return;
14            if (Catalogo_amIOnGoogle ())
15            {
16               var wcd = window.content.document;
17               var query = Catalogo_addWidgetToGoogle ();
18
19               if (query )
20               {
21                 Catalogo_basicSearchCompleted = false;
22                 var Catalogo_form = wcd.getElementById('CatalogoWidget_expand');
23                 Catalogo_OPACbasicSearch (Catalogo_form, query, wcd);
24                 var comm = wcd.getElementById('CatalogoWidget_submit');
25                 if(comm)
26                 {
27                    comm.addEventListener("click", function()
28                        {Catalogo_ExpandQuery(wcd, query); }, true);
29                 }
30               }
31            }
32         }
33
34 }
```

Fig 5.7 : Inizializzazione del browser e avvio dell'estensione nel file `catalogo.js`.

Come detto, una volta che si è verificato di essere in una pagina dei risultati di Google tramite la funzione `Catalogo_amIOnGoogle ()` , viene caricato lo scheletro dell'estensione tramite la funzione `Catalogo_addWidgetToGoogle ();` entrambe queste funzioni fanno parte del file `google_processor.js`.

5.2.2 File `google_processor.js`

Il file `google_processor.js` contiene le funzioni pensate per interagire con le pagine di Google caricate nel browser. In dettaglio contiene due funzioni:

- `Catalogo_amIOnGoogle ()`: serve per verificare, esaminando l'URL della pagina, se questa corrisponde ad una pagina di risultati del motore di ricerca;
- `Catalogo_addWidgetToGoogle ()` : serve per stampare lo scheletro del codice HTML dove l'estensione inserirà i dati prelevati dal Catalogo dell'OPAC del sistema bilbliotecario padovano.

Quando l'estensione carica lo scheletro HTML costruisce con le funzioni viste prima una serie di nodi che verranno poi riempiti con i risultati della ricerca fatta nel Catalogo del sistema bilbliotecario padovano. La funzione `Catalogo_addWidgetToGoogle ()` per prima cosa verifica se lo scheletro già esiste; se esiste ha già terminato, altrimenti procede con la creazione del sottoalbero relativo all'estensione. Costruisce un primo nodo di tipo `<div>` che conterrà tutti i dati prodotti dall'estensione e che fino al momento in cui i risultati non vengono visualizzati mostra un messaggio di attesa. A questo punto crea tre nodi figli sempre di tipo `<div>` che conterranno rispettivamente l'intestazione dell'estensione, i risultati da lei prodotti e dei link relativi all'Università. I nodi di tipo `<div>` sono dei contenitori di codice html che permettono di collocare testi, immagini o altro in zone predeterminate di una pagina web. In questo caso il primo nodo `<div>` creato permette di produrre una finestra, sulla destra dei risultati di Google, sovrapposta ai link sponsorizzati, che contiene i risultati dell'OPAC. Per la fase relativa alla query expansion sono necessari altri due nodi entrambi figli del secondo nodo `<div>` del secondo livello, quello relativo ai risultati. I due nuovi nodi sono di tipo diverso dai precedenti; il primo è del tipo `<form>` e servirà per contenere i risultati dell'OPAC ciascuno con la propria checkbox per registrare i giudizi di rilevanza dell'utente, il secondo è un nodo di tipo `<input>` relativo al tasto submit che permetterà di eseguire la query expansion sulla base dei risultati registrati nel nodo `<form>`.

La funzione una volta creati i nodi li collega come in figura 5.8, poi recupera il nodo relativo ai link sponsorizzati e lo elimina rimpiazzandolo con il primo nodo `<div>` dell'estensione. Una volta inserito in questo modo lo scheletro nella pagina recupera l'indirizzo della pagina dei risultati di Google per estrarre la query che verrà poi usata per fare la ricerca nel Catalogo dell'OPAC.

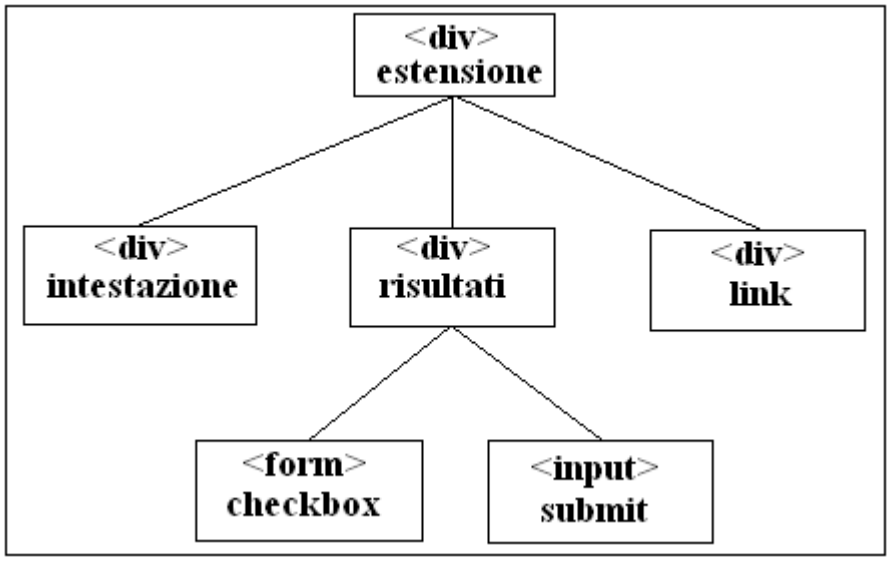


Fig: 5.8: La struttura ad albero dei nodi dell'estensione.

Una volta recuperata la query si procede con la vera ricerca nell'OPAC grazie alla funzione `Catalogo_OPACbasicSearch` del file `opac_search.js`.

5.2.3 File `opac_search.js`

Il file `opac_search.js` contiene tutte le funzioni necessarie per effettuare la ricerca nell'OPAC e visualizzarne i risultati. Nello specifico le funzioni sono:

- `Catalogo_OPACbasicSearch` serve per eseguire la ricerca della query di Google nel Catalogo dell'OPAC;
- `Catalogo_OPACparseOutput` serve per elaborare l'output del Catalogo in modo da estrarre i dati da visualizzare nella finestra dell'estensione;
- `Catalogo_context` è una funzione "di servizio" che elabora i dati dell'OPAC per collegarli poi ai checkbox che serviranno per eseguire il processo di query expansion.

Dopo aver stampato lo scheletro HTML dell'estensione ed aver recuperato la query di Google, si procede quindi con la ricerca nell'OPAC. Nel file `catalogo.js` viene quindi chiamata la funzione `Catalogo_OPACbasicSearch (element, query, wcd)`. I parametri della funzione sono: l'elemento dello scheletro HTML che conterrà i risultati della ricerca (nodo `<form>`), la query da ricercare nel Catalogo dell'OPAC e un riferimento alla pagina contenuta nel browser per eventuali modifiche dei suoi nodi.

```
1  function Catalogo_OPACbasicSearch (element, query, wcd)
2  {
3     var url = 'http://catalogo.unipd.it/F?func=find-e&find_scan_code=FIND_WRD
4               &find_scan_code_advanced=FIND_WRD&request='+ query;
5     var req = new XMLHttpRequest ();
6     req.onreadystatechange = function () {
7        if (req.readyState == 4)
8        {
9           if (req.status == 200) /
10          {
11             var codeToPrint = '';
12             codeToPrint = '<b>' + Catalogo_languageData.resultsHeader1 +
13                           ' <i><a style="color:#00C" href="' + url + '">' +
14                           unescape(query.split('+').join(' ')) + '</a></i> ' +
15                           Catalogo_languageData.resultsHeader2 + ':</b><br />';
16             var text = req.responseText;
17             codeToPrint += Catalogo_OPACparseOutput (url, text, 10, wcd);
18             element.innerHTML = codeToPrint;
19             Catalogo_basicSearchCompleted = true;
20             Catalogo_removeWaitMessage ();
21          }
22       }
23    };
24    req.open ('GET', url, true);
25    req.send (null);
26 }
```

Fig 5.9: Codice per la ricerca nell'OPAC.

La funzione per prima cosa compone l'indirizzo URL al quale richiedere i risultati dell'OPAC per la query di Google (riga 3). Per recuperare tali risultati è necessario usare una richiesta AJAX (Asynchronous JavaScript And XML); questa richiesta viene eseguita alle righe 24 e 25 dopo che è stato creato l'oggetto che la rende possibile (riga 5). Quando la richiesta cambia di stato si entra nella funzione della riga 6. L'elaborazione dei risultati della ricerca nell'OPAC sarà possibile solo nel caso in cui la richiesta raggiunga l'ultimo stato (riga 7) e, soprattutto, sia andata a buon fine (riga 9). Una volta verificato tutto questo, il codice sorgente della pagina

dei risultati OPAC, viene elaborato, chiamando la funzione `Catalogo_OPACparseOutput (url, text, limit, wcd)` (riga 17), per estrarne le informazioni da inserire nello scheletro HTML. Una volta recuperate tutte queste informazioni, l'elemento dello scheletro HTML, di tipo `<form>`, viene riempito grazie alla proprietà `innerHTML` (riga 18) e il nodo contenente il messaggio di attesa viene rimosso (riga 20).

La funzione per l'elaborazione del codice sorgente necessita di quattro parametri:

- l'indirizzo URL della pagina dei risultati OPAC che servirà come link nel caso ci sia un solo risultato;
- il codice sorgente della pagina dei risultati del Catalogo OPAC da elaborare;
- il numero di record da visualizzare nella finestra dell'estensione, nel caso ci siano più risultati;
- un riferimento alla pagina contenuta nel browser per le modifiche dei suoi nodi nel caso ci sia un risultato singolo.

In tutte le operazioni eseguite in questa funzione si fa un massiccio uso di funzioni per l'elaborazione delle stringhe. Il codice sorgente a disposizione è codice HTML, per cui la prima operazione necessaria è quella di rimuovere i commenti. Successivamente si identifica la porzione di codice contenete i risultati; questa sezione si presenta in due modi diversi nel caso di un risultato singolo e nel caso di risultati multipli, per cui si procede in due modi diversi a seconda del caso.

Nel caso di un risultato singolo si identifica il pattern che contiene le informazioni utili e si inserisce tale pattern nel codice HTML che verrà visualizzato dall'estensione. In questo caso, come già detto, non viene data la possibilità di procedere con l'espansione della query, per cui bisogna recuperare il nodo relativo al tasto submit per disabilitarlo.

```
1 var comm = wcd.getElementById('CatalogoWidget_submit');
2                 comm.disabled = true;
```

Fig 5.10: Codice per disabilitare il tasto nel caso singolo.

Nel caso di risultati multipli vengono identificati tutti i dati relativi ad ogni record, sempre grazie alle funzioni per l'elaborazione delle stringhe, ed inserendoli nel codice HTML da visualizzare, viene posto a fianco di ciascun record un elemento checkbox, ad esso collegato, per registrare i giudizi di rilevanza dell'utente. Per

colegare ciascun checkbox al suo record viene chiamata la funzione di servizio `Catalogo_context (title)` che permette di definire il nome di ciascun checkbox creando un legame con il record a cui appartiene.
In entrambi i casi, singolo o multiplo, la funzione `Catalogo_OPACparseOutput` restituisce il codice HTML da inserire nella finestra dell'estensione.
Una volta visualizzati i risultati, come detto, l'estensione, resta in attesa di un evento di tipo `click` sul tasto submit per tentare di espandere la query iniziale attraverso la funzione `Catalogo_ExpandQuery` del file `expansion.js` .

5.2.4 File `expansion.js`

Il file `expansion.js` è l'ultimo file dell'estensione e contiene tutte le funzioni che permettono di eseguire praticamente il processo di espansione della query; in particolare le funzioni presenti nel file sono:

- `Catalogo_ExpandQuery` serve per estrarre i giudizi di rilevanza dell'utente dal form dell'estensione e più precisamenti dai checkbox;
- `Catalogo_Operation` è la funzione che compie effettivamente il processo di "query expansion";
- `Catalogo_entry` è il costruttore di un oggetto per un termine del contesto estratto dai dati dei checkbox e comprende un campo per il termine stesso (`term`) e due relativi alla frequenza nei documenti giudicati rilevanti (`weigth_rel`) ed in quelli giudicati non rilevanti (`weigth_tot`);
- `Catalogo_obj` è il costruttore di un oggetto per un termine candidato all'espansione della query e comprende un campo per il termine (`index`) e uno per il suo peso (`points`).

Come già accennato l'estensione contiene un tasto submit che permette di espandere la query iniziale di Google. Una volta visualizzati i risultati di ricerca dell'OPAC, l'utente è invitato a selezionare le opere che ritiene interessanti; quando esegue un click sul tasto dell'estensione viene chiamata in causa la funzione `Catalogo_ExpandQuery (element, query)`, i cui parametri sono l'elemento dell'estensione che contiene i dati espressi dall'utente e la query iniziale

da espandere. Questa funzione recupera i valori dei checkbox e i dati, relativi a titolo e autori, ad essi associati suddividendoli in quattro gruppi: titoli rilevanti, autori rilevanti, titoli non rilevanti e autori non rilevanti per pesarli poi con costanti diverse nel processo di pesatura. Si ritiene in questo caso che titoli ed autori devano essere considerati in maniera diversa in quanto il contesto di un'opera viene rappresentato in primo luogo dall'argomento che tratta e quindi dal titolo, e poi in secondo luogo dagli autori che l'hanno realizzata. In questa funzione si controlla anche se l'utente non ha espresso alcun giudizio; in tal caso viene avvisato con un avviso sullo schermo. Una volta definiti i quattro gruppi, viene chiamata la funzione principale di questa fase dell'estensione: `Catalogo_Operation (context, query, element, basket, R, check, good_names, bad_names)` con i seguenti parametri:

- l'insieme dei titoli considerati rilevanti;
- la query da espandere;
- un riferimento alla pagina caricata nel browser;
- l'insieme dei titoli non considerati rilevanti;
- il numero di opere considerate rilevanti;
- il numero totale di opere presentate dall'estensione;
- l'insieme degli autori delle opere considerate rilevanti;
- l'insieme degli autori delle opere non considerate rilevanti.

La funzione comincia ad organizzare i quattro insiemi (soprattutto i due relativi ai titoli) seguendo le fasi del processo di indicizzazione tipico del reperimento dell'informazione. Il primo punto prevede l'eliminazione delle *stop word*. La nostra *stop list* è costituita da articoli, preposizioni e avverbi, nelle due lingue (italiano ed inglese), ed è contenuta in un semplice file di testo, `stoplist.txt`. Il formato del file prevede che ci sia una *stop word* per ogni riga. Per leggere un file di testo diverso da HTML, in JavaScript non sono necessarie classi o funzioni particolari; si può implementare la stessa struttura AJAX usata per leggere le pagine dell'OPAC, con la sola differenza che il campo `status` dell'oggetto di richiesta non conterrà lo stato http del suo esito, ma avrà valore 0 se la lettura del file è andata a buon fine (riga 6). Il contenuto del file viene quindi letto (riga 7) e utilizzato per eliminare le *stop word*

dai termini presenti nei due insiemi dei titoli (righe 11-20, uguali per l'insieme dei titoli non rilevanti), visto che gli insiemi degli autori sono costituiti da soli nomi e cognomi.

```
1  function Catalogo_Operation (context, query, element, basket,
2                               R, check, good_names, bad_names)
3  {
4     var req = new XMLHttpRequest ();
5     req.onreadystatechange = function () {
6                  if (req.readyState != 4 || req.status != 0) return;
7                  var stoplist = req.responseText;
8                  var j = 0;
9                  var list = new Array();
10
11                 for( var i = 0; i < context.length; i++)
12                 {
13                         var word = new RegExp(context[i], "gi");
14                         var  stopword = stoplist.match(word);
15                         if(!stopword)
16                         {
17                                 list[j] = context[i];
18                                 j= j+1;
19                         }
20                 }
21
22
23                 //  .....
24
25                 // prosegue codice per la pesatura dei termini
26    };
27    req.open ('GET', 'chrome://catalogo/content/stopword/stoplist.txt', true);
28    req.send (null);
29 }
```

Fig 5.11: Codice per eliminare le stop word.

A questo punto, i termini che compongono gli insiemi 'titoli rilevanti' ed 'autori rilevanti' sono i candidati per espandere la query iniziale; la fase successiva prevede la loro pesatura considerando però anche i termini degl'altri due insiemi 'titoli non rilevanti' ed 'autori non rilevanti'. La prima cosa da fare è rilevare le frequenze di ciacun termine nei vari insiemi tramite l'uso di array associativi; poi per ogni termine dei documenti rilevanti creare un oggetto contenente le seguenti informazioni:

- termine considerato;
- frequenza del termine nei documenti considerati rilevanti;
- frequenza del termine nei documenti visualizzati dall'estensione;
- costante utile nella fase di pesatura.

```
1  function Catalogo_entry(term, weigth_rel, weigth_tot, w)
2  {
3     this.term = term;
4     this.weigth_rel = weigth_rel;
5     this.weigth_tot = weigth_tot;
6     this.w = w;
7     this.addWeigth = function(new_weigth) {
8        this.weigth_tot += new_weigth;
9     }
10    this.toString = function () {
11       return ' ' + term + ' : ' + weigth_rel + ' / ' +
12                    weigth_tot + ' * ' + w;
13    }
14 }
```

Fig 5.12: Funzione per creare oggetti contenenti i termini e le loro frequenze.

Tutti gli oggetti, di questo tipo, costruiti in questa fase vengono riuniti in un unico array. Per ogni oggetto contenuto nell'array si procede a creare un nuovo tipo di oggetto, questa volta necessario per costruire il dizionario da cui estrarre poi i termini da usare per espandere la query. In ogni istanza del nuovo oggetto vengono inseriti:

- il termine indice candidato all'espansione;
- il peso di questo termine secondo lo schema di pesatura usato;

```
1  function Catalogo_obj(index, points)
2  {
3          this.index = index;
4          this.points = points;
5          this.toString = function () {
6                  return ' ' + index + ' : ' + points ;
7          }
8  }
```

Fig 5.13 : Funzione per creare una voce del dizionario.

Per quanto riguarda la pesatura del singolo termine si è deciso di ricorrere allo schema *wpq* che, come dimostrato dallo studio di Efthimiadis (1995), risulta essere il più efficace per quanto riguarda il processo di query expansion. La fomula, come visto nel capitolo precedente, usa, al suo interno, un'altra formula di pesatura che è quella dell'algoritmo *f4*, che permette in questo modo di privilegiare i termini rari cioè quelli più rappresentativi per il contesto. I pesi risultanti poi, dall'applicazione della formula *wpq*, vengono rielaborati applicando una costante di pesatura che permette di dare maggior peso ai termini dei titoli, rispetto agli autori.

Nella figura si vede come alle righe 2-5 venga applicata la formula *f4*, successivamente completata secondo lo schema *wpq* (riga 6).

```
1  var f4 = 0;
2  f4 = (order_score[i].weigth_rel + 0.5)*
3      (check - order_score[i].weigth_tot - R + order_score[i].weigth_rel + 0.5);
4  f4 = f4/((order_score[i].weigth_tot - order_score[i].weigth_rel + 0.5)
5              *(R - order_score[i].weigth_rel + 0.5));
6  var wpq = Math.log(f4)*((order_score[i].weigth_rel/R)-
7          ((order_score[i].weigth_tot - order_score[i].weigth_rel)/(check - R)));
8  if (order_score[i].weigth_tot == check)
9      { wpq += (order_score[i].weigth_rel/check); }
10 wpq = order_score[i].w * wpq;
11 index_term.push( new Catalogo_obj(order_score[i].term, wpq));
```

Fig 5.14 : Codice relativo alle fasi di pesatura dei termini.

Alla riga 11 viene creato l'oggetto del dizionario relativo al termine in considerazione. Quando tutti i termini candidati per l'espansione sono contenuti nel dizionario, questo viene ordinato, secondo il peso dei singoli termini, in ordine decrescente.

Una volta ottenuto il dizionario ordinato, si comincia dal primo termine contenuto in esso e si procede nel selezionare quei termini che si trovano nelle prime posizioni del dizionario e che hanno un peso pari al peso più alto rilevato. La scelta dei termini da usare per espandere la query ricade su quelli che non compaiono già nella query iniziale e che, per l'appunto, hanno il peso più alto.

```
1  var plus = false;
2  var i = 0;
3  query = query.split('+').join(' ');
4  var max = index_term[0].points;
5  while ((index_term[i]) && (index_term[i].points == max))
6  {
7     var exp = new RegExp(index_term[i].index, "gi");
8     var control = query.match(exp);
9     if (!control)
10    {
11       query += ' ' + index_term[i].index;
12    }
13    i++;
14 }
```

Fig 5.15: Codice per la selezione termini dal dizionario.

A questo punto la nuova query è costruita e bisogna sottoporla al motore di ricerca Google. Per fare questo è necessario recuperare un riferimento alla casella di testo, del motore di ricerca stesso, per l'immissione della query. Grazie a DOM Inspector si

è scoperto che, la casella di testo, è il primo elemento del nodo `<form>` di nome *q*. Per questo si usa la funzione `getElementsByName()` per recuperare i riferimenti agli elementi della pagina di nome *q* (riga 1), poi si inserisce la nuova query nel primo riferimento, che è quello relativo alla casella di testo (riga 3).

```
1  var text_query = element.getElementsByName("q");
2  var submit_query= element.getElementById('tsf');
3  text_query[0].value = query;
4  submit_query.submit();
```

Fig 5.16: Codice finale per sottomettere la nuova query al motore di ricerca.

Per completare il lavoro è necessario fare in modo che Google raccolga la nuova query ed elabori i dati. Per fare ciò è necessario recuperare un riferimento al nodo `<form>` che contiene la casella di testo; ancora una volta, grazie a DOM Inspector, si è scoperto che il nodo ha id *tsf*. Recuperato, quindi, il riferimento al nodo (riga 2), si fa eseguire l'operazione `submit()` (riga 4) che invia i dati della casella di testo a Google eseguendo una nuova ricerca con la nuova query.

Questo processo avvia anche una nuova ricerca nell'OPAC, con la nuova query, creando così un ciclo nell'estensione che si interrompe solo nel caso di query non più epandibili o nel caso l'OPAC fornisca un solo risultato.

5.3 Le scelte

La prima parte dell'estensione, che prevede il collegamento all'OPAC e la visualizzazione dei risultati di ricerca, si è basata principalmente sulle funzioni di elaborazione delle stringhe, infatti gli unici problemi si sono verificati nell'estrarre, dal codice sorgente della pagina dei risultati OPAC,, i dati da presentare all'utente. Nel caso in cui l'OPAC restituisca un risultato solo, nel link, dell'estensione, che permette di collegarsi alla pagina OPAC relativa al record si è scelto di visualizzare solo il titolo dell'opera e gli autori, in quanto spesso la pagina del singolo record varia per disposizione dei dati rendendo difficile l'identificazione di un pattern standard.

La parte più problematica, nonché la più innovativa, dell'estensione è stata quella relativa alla query expansion. Per prima cosa si è scelto di far intervenire l'utente nel processo di relevance feedback attraverso l'uso dei checkbox. Assumendo i risultati

OPAC di ottimo valore si è deciso di lavorare su questi per estrarre termini utili a definire in modo migliore l'esigenza informativa dell'utente.
Per passare i dati all'applicazione ed estrarre i giudizi di rilevanza dai checkbox è stato necessario modificare la struttura iniziale dell'estensione inserendo all'interno del nodo `<div>`, originariamente creato per raccogliere i risultati OPAC, un nodo `<form>`. Il nodo `<form>` all'inizio creava problemi perché, inserito nella pagina del motore di ricerca, cercava di passare i dati a quest'ultimo, che non li riconosceva; per evitare tutto ciò è stato necessario bloccare il nodo in modo tale che non passasse i dati alla pagina di Google e creare sullo stesso livello un nodo `<input>`, contenente un tasto, che permettesse di controllare le richieste dell'utente. Il costante controllo sul nodo `<input>` permette di rilevare quando l'utente preme il tasto per espandere la query.
La fase centrale del processo di query expansion è la pesatura dei termini. Per prima cosa è stato necessario suddividere i titoli delle opere dagli autori per pesare le due categorie in mdo tale da dare più importanza ai termini dei titoli che rappresentano più propriamente il contesto dell'opera. Successivamente, per quanto riguarda lo schema di pesatura da adottare si era partiti con un banale *TF*; questo schema aveva portato ad un peso unitario di ogni termine, nella maggior parte dei casi. Con *TF*, nella quasi totalità dei casi, l'estensione prelevava la prima parola del titolo e la inseriva nella query, fornendo così un risultato non adeguato alle aspettative. A questo punto si è ricorso agli schemi di pesatura inventati appositamente per la query expansion, partendo con l'algoritmo *f4*. Questa scelta aveva prodotto, da subito, buoni risultati per quanto riguarda i termini da aggiungere alla query, soprattutto nel caso in cui l'utente segnalasse un unico record come rilevante; il primo caso che ha sollevato dei dubbi è stato quando durante le prove si è inserita la query 'massimo melucci' e tra i risultati sono stati scelti due record:

- String Processing and Information Retrieval / Alberto Apostolico, Massimo Melucci;
- Ipertesti e information retrieval / Mario Ricciardi, Maristella Agosti, Massimo Melucci;

e l'estensione ha fornito come termini aggiuntivi 'string processing' e 'ipertesti', quando in realtà era più logico attendersi, come risultato 'information retrieval', che è l'argomento comune tra i due record considerati rilevanti.

Per favorire situazioni simili si è ricorsi allo schema di pesatura *wpq*, che fa comunque uso dell'algoritmo *f4*, ma che in questi casi, fornisce risultati più vicini alle attese per quanto riguarda i termini scelti pesando maggiormente termini più frequenti. Lo schema *f4*, come visto, favorisce i termini rari, mentre con *wpq* si considerano in proporzione maggiore anche termini con frequenza più alta. Lo schema *wpq* è stato dimostrato essere uno dei più validi, se non il migliore in generale [Eft95], ed anche in questo caso particolare, sembra essere l'unico schema a fornire risultati adeguati.

Per quanto riguarda il fatto di non permettere di espandere la query nel caso in cui l'OPAC fornisca un solo risultato; è stata una scelta dovuta al fatto che se la query ha permesso di identificare, con precisione, un unico risultato OPAC, ricordando che l'estensione prende come riferimento i dati dell'OPAC, significa che è già una query di buon livello.

6 Conclusioni

Questo lavoro di tesi ha comportato un grosso lavoro di studio soprattutto nella prima parte, riguardante l'online advertising ed il click fraud. Per quanto riguarda l'estensione sviluppata è stato necessario studiare come si realizzano le estensioni per un browser come Mozilla Firefox e fare pratica con JavaScript. La parte più difficile è stata senza dubbio la scelta dello schema di pesatura ed il suo perfezionamento per eseguire l'espansione della query secondo le attese. Lo studio della letteratura ha permesso di trovare uno schema adeguato alla situazione dopo vari tentativi andati a vuoto. L'estensione sembra completare le sue operazioni in un tempo abbastanza breve (è pur sempre un applicazione client).

Nella versione attuale l'estensione da risultati soddisfacenti sia nel caso si scelga un solo record, come rilevante, sia nel caso si scelgano più record. I risultati sono buoni e rispettano le attese permettendo di ottenere migliori risultati di ricerca in Google. È molto difficile che nei termini scelti per espandere la query venga scelto un autore perché normalmente l'argomento di interesse è espresso dal titolo e non dagli autori e per questo si usano delle costanti di pesatura diversa.

Per quanto riguarda i possibili sviluppi futuri si parla di creare funzioni simili a quelle già realizzate in questo caso però adattate a motori di ricerca diversi da Google, oppure che usino dati provenienti da altri cataloghi OPAC.

Se si desidera fare lavorare quest'estensione su motori di ricerca divarsi da Google è necessario riscrivere le funzioni del file `google_processor.js` e modificare leggermente la funzione `Catalogo_Operation` del file `expansion.js`.

Per quanto riguarda il file `google_processor.js`, sarà necessario sostituirlo con un file che contiene:

- un metodo per verificare se la pagina caricata nel browser è una pagina dei risultati del motore di ricerca considerato;
- un metodo che crei materialmente la finestra dell'estensione aggiungendola alla pagina del motore di ricerca nella posizione desiderata;

mentre per quanto rigurda il file `expansion.js` sarà necessario trovare i giusti riferimenti per sottoporre la query estesa al motore di ricerca.

Nel caso, invece, si desiderasse usare dati provenienti da altri cataloghi OPAC, le modifiche principali da fare sono:

- cambiare l'url nella funzione `Catalogo_OPACbasicSearch` con quello della pagina del nuovo catalogo;
- studiare il codice HTML delle pagine dei risultati dell'OPAC e modificare la struttura della funzione `Catalogo_OPACparseOutput` applicando le giuste funzioni di elaborazione delle stringhe per estrarre i dati dal codice sorgente delle pagine.

Per la fase di espansione della query l'unico sviluppo possibile è quello di cercare di usare uno schema di pesatura diverso che però può cambiare drasticamente i risultati del processo.

Nella fase di elaborazione dei dati per espandere la query è anche possibile utilizzare un stoplist più completa inserendola nel file `stoplist.txt`.

Appendice

A1 La stoplist

adesso	agli	ai	al	alla	allo
allora	altre	altri	altro	anche	ancora
avere	aveva	avevano	ben	buono	che
chi	cinque	come	comprare	con	consecutivi
consecutivo	cosa	cui	da	dal	del
della	dello	dentro	deve	devo	di
doppio	due	ecco	fare	fine	fino
fra	gente	giù	ha	hai	hanno
ho	il	indietro	invece	io	la
lavoro	le	lei	lo	loro	lui
lungo	ma	me	meglio	molta	molti
molto	nei	nella	no	noi	nome
nostro	nove	nuovi	nuovo	oltre	ora
otto	peggio	per	però	persone	più
poco	primo	promesso	qua	quarto	quasi
quattro	quello	questo	qui	quindi	quinto
rispetto	sarà	secondo	sei	sembra	sembrava
senza	sette	sia	siamo	siete	solo

sono	sopra	soprattutto	sotto	stati	stato
stesso	su	subito	sul	sulla	tanto
te	tempo	terzo	tra	tre	triplo
ultimo	un	una	uno	va	vai
voi	volte	vostro	about	an	and
are	as	at	be	by	com
for	from	how	in	is	it
of	on	or	that	the	this
to	was	what	when	where	who
will	with	through	www		

Elenco delle figure

Bibliografia

[07Fe] David Grossman: Relevance Feedback; presentazione www.ir.iit.edu/~dagr/cs529/files/handouts/07Feedback.pdf

[AdBr] Guida per l'advertiser, AdBrite http://www.adbrite.com/mb/how-advertisers-guide.php

[AdPu] Guida per il publisher, AdBrite https://www.adbrite.com/mb/how-publishers-guide.php

[Bad] Gandhi, Jakobsson, Ratkiewicz: Badvertisements: Stealthy Click-Fraud with Unwitting Accessories; Journal of Digital Forensic Practice, 2006

[Cat] Catalogo del sistema bibliotecario padovano; http://catalogo.unipd.it/F/?func=file&file_name=find-b

[Ca_Gu] Guida all'uso; Catalogo del sistema bibliotecario padovano; http://catalogo.unipd.it/F/?func=file&file_name=help-1

[Clab] Bloch, Eroshenko: How To Defend Your Website Against Click Fraud; Clicklab; 2004.

[ClF] Jansen: Click Fraud, Computer 2007 IEEE

[DaCi] Davide Cisco: Un'estensione di un browser per l'accesso alle collezioni delle biblioteche nazionali europee basata su metodi di pubblicità web; Università degli Studi di Padova; 2008/2009.

[DOM] S.Wilsher : DOM Inspector; programma 2008, https://addons.mozilla.org/it/firefox/addon/6622

[Eft92] Efthimis N. Efthimiadis: Interactive query expansion and relevance feedback for document retrieval systems. City University, London,1992.

[Eft93] Efthimis N. Efthimiadis: A user-centered evaluation of ranking algorithms for interactive query expansion; SIGIR 1993

[Eft95] Efthimis N. Efthimiadis : User Choices: A New Yardstick for the Evaluation of Ranking Algorithms for Interactive Query Expansion. Information Processing & Management; 1995.

[EQE] Efficient query expansion; B. Billerbeck; RMIT University, Melbourne, Victoria, Australia; September 2005.

[GoAd] AdWords Google; sito web http://adwords.google.com/support/aw/bin/

[Gog] Guida Google sull'advertising; sito web http://www.googleguide.com/ads.html

[GoSe] AdSense Google; sito web https://www.google.com/adsense/login/it/

[GoWo] Statistiche di AdWords Google; sito web https://www.google.com/intl/it_it/adwords/select/news/sa_mar04.html

[IASel] Edelman, Ostrovsky, Schwarz: Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords, American Economic Review, 2007

[IDC 2008] International Symposium on Intelligent Distributed Computing

[IIR] Introduction to Information Retrieval; C.D. Manning,P. Raghavan, H. Schütze; Cambridge University Press 2008

[JaSc] JavaScript Passo per Passo di Steve Suehring; Mondadori Informatica; 2008

[Off07] blog ufficiale di Google AdWords: Invalid clicks – google's overall numbers; http://adwords.blogspot.com/2007/02/invalid-clicks-googles-overall-numbers.html, Febbraio 2007

[OnAF] Daswani, Mysen, Rao, Weis, Gharachorloo, Ghosemajumder e the Google Ad Traffic Quality Team: Online Advertising Fraud del libro

Crimeware: understanding new attacks and defenses, First edition, Addison-Wesley Professional 2008

[POAd] Shanahan: Online Advertising Business Models, Technologies and Issues: From “Mad Man” to Wall Street; ESSIR 2009

[PPPe] Goodman: Pay-Per-Percentage of Impressions: An Advertising Method that is Highly Robust to Fraud; ACM E-Commerce’05 Workshop on Sponsored Search Auctions

[ReIn] Reperimento dell’informazione, Information retrieval concetti, architetture e modelli dei motori di ricerca; M. Agosti, M. Melucci; a.a. 2007/2008 (seconda edizione).

[SSA] Aggarwal, Feldman, Muthukrishnan, Pál : Sponsored Search Auctions with Markovian Users; WINE 2008.

[SEn] Search Engines: Information retrieval in practice; B. Croft, D. Metzler, T. Strohman; Addison Wesley 2009.

[SecP] Mayer, Nissim, Pinkas, Reiter: On the Security of Pay-Per-Click and Other Web Advertising Schemes; Computer Networks, Volume 31, 1999

[Sci] Sponsored Search; sito web; http://www.scitopics.com/Sponsored_Search.html

[TuEx] Tutorial per estensioni; sito web http://devilsworkshop.org/shortest-tutorial-for-firefox-extensiontoolbar-development/

[Wik] Wikipedia; sito web http://en.wikipedia.org/wiki/

[Will] Come lavora un’ad network; sito web http://willscullypower.wordpress.com/2008/11/19/how-do-an-ad-exchange-and-an-ad-network-really-work/

[Yah] Yahoo Advertising; sito web http://help.yahoo.com/l/en-it/yahoo/ysm/sps/screenref/71984.html