Universitat
Autònoma
de Barcelona

5266: DisPA, Un Agent per a la Cerca Privada a la Web

Memòria del Projecte de Fi de Carrera
d'Enginyeria en Informàtica
i Grau en Matemàtiques.
Realitzat per *Marc Juárez*
i dirigit per *Guillermo Navarro* i *Vicenç Torra*

Bellaterra, 29 de gener de 2013

Universitat Autònoma de Barcelona

escola d'enginyeria

Els sotasignats, Guillermo Navarro i Vicenç Torra

CERTIFIQUEN:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Marc Juárez.

I per tal que consti firmen la present.

Signat:

Bellaterra, 29 de gener de 2013

A la Cesca, al Mario i a la Clara,
els pilars de la meva vida.

Agraïments

El meu primer agraïment va dirigit a en Vicenç Torra, per haver confiat en mi i haver-me contractat pel lloc de feina en el qual he portat a terme aquest projecte. Aquesta decisió ha permès iniciar-me en el món de la recerca i fer ús dels coneixements que he adquirit durant la carrera. Tant ell com en Guillermo Navarro es van oferir per dirigir aquest projecte i els seus consells m'han estat molt útils per aprendre i desenvolupar-me en aquest nou món.

Durant aquest any a l'IIIA he gaudit d'una companyia immillorable i de la qual només em queden bones paraules. Vull agrair a tots els companys que han col·laborat en el projecte contestant enquestes i registrant queries. En especial a Daniel Abril, Dave De Jonge, Jordi Marés i Sergi Martínez, pels seus ànims i per haver-me donat un cop de mà sempre que l'he necessitat.

També he comptat amb el suport la meva família, els meus pares i la meva germana, fent-me costat tant en els bons moments com en els no tant bons. Sense oblidar-me de l'Aina, perquè sense la seva paciència i els seus ànims avui no podria escriure aquestes línies.

M'agradaria fer especial menció a tots els companys amb els que he compartit aquests anys de carrera i he viscut tants bons moments, sobretot al Solanas, l'Adrià, el Joan Enric, el Lluís, el Manolo, la Sílvia i el Ricard.

Per concloure, vull recordar que aquest projecte tampoc hagués estat possible sense totes les persones que han col·laborat en la meva formació durant aquests anys.

Índex

Capítol 1

Introducció

Avui en dia, els motors de cerca de la Web juguen un paper molt important a Internet. Per tal de fer-nos una idea, segons l'últim informe global que ha publicat *comScore*, que data del 2009, es fan 29 milions de cerques per minut a tot el món (comScore, 2009). Com podem imaginar, aquesta xifra suposa una aportació de trànsit de dades a la xarxa molt superior al de la majoria de llocs web. En el mateix estudi es fa referència al mercat de cerques que, aquell any, va augmentar un 46% i estava repartit entre els principals motors de cerca de la següent manera: Google (67.5%), Yahoo! (7.8%), Baiduu (7%) i Bing (7%). Fins i tot hi ha fonts que afirmen que actualment Google ja controla vora el 80% del *share* mundial i en conseqüència, s'erigeix com el cercador per excel·lència a tot el món (Netmarketshare, 2012; statOWL, 2012).

Els motors de cerca recullen informació dels usuaris i de les seves cerques amb diverses finalitats. D'una banda, volen explotar aquestes noves possibilitats de negoci; així, poden portar a terme tasques d'investigació de marketing (*Marketing Research*) i oferir publicitat personalitzada (*Target Advertising*). D'altra banda, el creixement exponencial de la Web ha deixat al descobert la necessitat de millorar el servei que ofereixen (Norvig, 2011). Per solucionar-ho, el més comú és utilitzar la informació que recullen a fi d'elaborar perfils d'usuari (*Profiling*). Aquesta tècnica consisteix a analitzar els registres del servidor, també anomenats *query logs* o *server logs*, on es recull la majoria d'activitats de l'usuari: termes que ha introduït, enllaços que ha clicat, adreça IP d'origen, identificadors de la sessió, entre d'altres. A partir d'aquí es dedueixen les seves principals àrees d'interès (e.g. futbol, informàtica, política, etc.), i algunes dades demogràfiques com l'edat, el sexe o la nacionalitat. Així, per oferir una cerca més personalitzada, els algorismes del cercador reordenen les llistes de resultats d'acord amb aquestes preferències (Speretta i Gauch, 2005). Malgrat tot, també cal tenir en compte els seus desavantatges. L'activista

Eli Pariser destaca, com a principal inconvenient d'aquests algorismes de personalització, un fenomen que anomena *Filtre bombolla*[1]. També alerta de la violació de la privadesa que comporta aquest rastreig dels usuaris, especialment quan es fa a través de diferents dominis d'Internet (Pariser, 2011). Val a dir que, malgrat la seva poca popularitat, també s'han desenvolupat motors de cerca alternatius que no registren cap dada com *DuckDuckGo* (Buys, 2010) o el recentment desaparegut *Scroogle* (WorldNetDialy, 2007).

A tot això s'ha d'afegir una creixent sensibilització dels usuaris envers la difusió de la seva informació a la xarxa (Whelan, 2005; Rambam, 2006; Acquisti i Gross, 2006), possiblement causada per l'expansió de les xarxes socials que s'ha produït en aquesta última dècada. Cada cop es penja més informació personal a Internet i això representa una amenaça per a la seva privadesa. En el cas dels cercadors d'Internet, el risc no s'aprecia tant directament i encara ara hi ha una gran quantitat d'usuaris que no tenen consciència dels perills que involucra el seu ús. Un cas històric en aquest sentit va tenir lloc el 4 d'agost de 2006: la companyia de serveis d'Internet *AOL*[2] va pujar a la seva web un arxiu que contenia vint milions de cerques que més de 650.000 usuaris havien fet en un període de tres mesos (Arrington, 2006). Tot i que els objectius de l'empresa eren ajudar a la comunitat de recerca en Recuperació de la Informació (*Information Retrieval*), uns periodistes del New York Times no van trigar a localitzar amb nom i cognoms un dels usuaris: Thelma Arnold, una dona que vivia a Lilburn, a l'estat d'EEUU de Georgia. Van realitzar la identificació només amb els termes que la dona cercava i l'ajuda d'una guia telefònica (Barbaro i Zeller, 2006). Com que aquesta dona cercava noms de familiars i establiments del seu poble, els periodistes van deduir que les cerques havien estat fetes per algun Arnold de Lilburn (només n'eren 14). Tot seguit, només els van haver de trucar un per un. L'arxiu va ser retirat pocs dies després de la seva alliberació, però ja havia estat descarregat i penjat en altres servidors (Sadetsky, 2006). Actualment, existeixen comunitats senceres que es dediquen a estudiar aquests *logs* i fer suposicions sobre les identitats dels usuaris (Search-logs, 2006; Titanium, 2006). Arran d'aquest escàndol, el cap del departament d'investigació d'AOL va dimitir i la companyia va haver de fer front a múltiples denúncies judicials (Mills, 2006; EFF, 2009). Aquests esdeveniments van tenir especial rellevància perquè, malgrat que els identificadors d'usuari dels registres publicats havien estat *anonimitzats*[3], quedava demostrat que és

[1] Descriu els algorismes de cerca com filtres que només deixen passar informació que agradarà a l'usuari. Aquests algorismes poden aïllar-lo dins la seva pròpia *bombolla* privant-li l'accés a opinions diferents a la seva i limitant la seva visió de la realitat.

[2] Abans coneguda com *America Online*.

[3] De l'anglès *anonimize*, es refereix a la manipulació de les dades per tal que preservin l'anonimitat dels figurants.

possible enllaçar un usuari amb una identitat real només utilitzant les seves consultes o, si més no, reduir l'espai de cerca fins a ser manejable. Així mateix, la informació continguda en aquestes cerques ja suposa un perill per a la privadesa de l'usuari, que pot considerar la consulta d'alguns temes com a privats. Exemples clars són les cerques sobre malalties o continguts pornogràfics.

Per les raons que acabem d'exposar, creiem que el desenvolupament de programari que protegeixi la privadesa dels usuaris a la Web és un tema d'actualitat, especialment si es tenen en compte tècniques com la personalització de la cerca.

1.1 Antecedents

El problema de la Recuperació Privada de la Informació (PIR, de l'anglès, *Private Information Retrieval*), va ser proposat teòricament per primer cop per Benny Chor (Chor *et al.*, 1995) i, l'any 1997, es va estudiar computacionalment per Kushilevitz i Ostrovsky (1997). Tracta la recuperació d'informació d'una base de dades sense el coneixement d'aquesta per part del servidor. Normalment, pel desenvolupament de protocols que resolguin el problema, la base de dades es modelitza com un vector i l'objectiu és mantenir privat l'índex de la posició on es troba l'element recuperat. L'algorisme més senzill que aconsegueix ocultar completament aquest índex requereix passar almenys un cop per totes les posicions de la base de dades. El principal inconvenient és la seva complexitat, que és $\mathcal{O}(n)$, on n és la longitud del vector i sol ser molt gran. Malgrat que treballs posteriors han trobat solucions amb complexitats logarítmiques (Gentry i Ramzan, 2005; Lipmaa, 2010), la majoria requereixen cooperació dels servidors que proporcionen la base de dades.

Recentment, s'ha traslladat el problema PIR a les cerques a Internet. En aquest context, l'usuari vol cercar informació sense que el proveïdor de la cerca conegui de què es tracta. Tanmateix, dins del marc del PIR tradicional sorgeixen una sèrie de dificultats:

1. La Web té estructura de graf i no es pot modelitzar com un vector.

2. La seva mida fa que els protocols PIR anteriors no es puguin aplicar a la pràctica.

3. No es pot suposar cooperació per part dels proveïdors de la cerca. La raó principal és que són reticents a afegir còmput "extra" en els seus algorismes.

En el cas d'AOL, cada nom d'usuari es va substituir per un nombre triat a l'atzar. Per exemple, el registre de Thelma Arnold va passar a ser el de l'usuari 4417749.

Un enfocament diferent al PIR és l'ús de mescladors digitals o *digital mixes*. Els *mixes* van ser inventats per David Chaum el 1981 (Chaum, 1981) per tal d'oferir serveis de correu electrònic anònim. Treballen a la capa de xarxa i fan que les comunicacions siguin difícils d'analitzar (e.g. conèixer l'origen o el destinatari). El funcionament es basa a encriptar el missatge per capes, com si d'una ceba es tractés, on la capa més interior és el missatge en sí. Aleshores, una sèrie de *proxies* encadenats van desencriptant, capa per capa, fins que arriba al destinatari. El resultat és que qui rep el missatge no coneix la IP d'origen, només la IP de l'últim *proxy*. Un dels protocols més famosos que segueix aquesta estratègia és el projecte ToR[4] (The ToR Project, 2002). L'enfocament que donen els mescladors es diferencia del dels PIR perquè, enlloc de protegir la informació del missatge, es protegeix la identitat de l'usuari a través d'emmascarar les metadades de la comunicació. No obstant, no solucionen el problema ja que, com hem vist a la introducció amb el cas de Thelma Arnold, el contingut de les cerques és suficient per identificar l'usuari.

Una altra manera d'atacar el problema és relaxar la condició de privadesa i protegir la informació amb una certa probabilitat de revelació. A la literatura trobem diferents estratègies en aquesta línia. Per exemple, dins del camp de Control de la Revelació d'informació (*Data Disclosure Control*, DDC), trobem solucions centrades en *anonimitzar* els registres de *queries*[5] (Cooper, 2008). També trobem sistemes que intenten preservar les propietats estadístiques (*Statistical Disclosure Control*, SDC) a la vegada que ofereixen algun grau de privadesa (e.g. compleixen la *k-anonimitat*[6] (Navarro-Arribas i Torra, 2009; Hong *et al.*, 2009)). Fins i tot n'hi ha que fan ús de taxonomies semàntiques per *microagregar*[7] de manera coherent amb els interessos dels usuaris (He i Naughton, 2009). En aquesta última branca de recerca s'ha emprat l'ontologia WordNet (Miller, 2010) i, més recentment, l'Open Directory Project (Erola *et al.*, 2011). Malauradament, aquests mètodes també assumeixen el vist i plau de la companyia que ofereix el servei perquè es faci un tractament dels seus *logs*.

Per altra banda, també existeixen protocols no cooperatius que s'han desenvolupat en aquests últims anys. A continuació farem una breu descripció de cadascun d'ells.

GooPIR (Domingo-Ferrer *et al.*, 2008): GooPIR és una generalització d'un

[4] Són les sigles de *The Onion Router*, ja que el funcionament del protocol recorda a com es pela una ceba.

[5] En aquest treball utilitzem freqüentment el terme *query* per referir-nos a la consulta introduïda al cercador i formulada mitjançant una conjunció de termes.

[6] És una propietat que poden complir els *datasets* on k registres són indistingibles (Sweeney, 2002; Samarati, 2001)).

[7] Tècnica que consisteix en reduir grups de registres a un sol. D'aquesta manera es fa més difícil identificar els usuaris dels registres *microagregats*, mentre que es mantenen les propietats estadístiques (mitjana, variància, etc).

protocol PIR tradicional. El seu funcionament es basa a afegir conjunts de termes falsos en disjunció exclusiva amb els originals. D'aquesta forma el cercador retorna, o bé resultats dels termes originals, o bé resultats dels termes afegits. Aleshores, per garbellar els resultats que ens interessen és tant fàcil com mostrar només aquells que contenen algun terme original. Amb la finalitat de reduir la probabilitat de detectar els termes falsos, es fa servir un diccionari amb una freqüència associada a cada terme. Primer es defineix un interval per a la freqüència a partir dels termes de la *query* i, seguidament, es trien termes del diccionari que es trobin en el mateix interval. A més a més, es crea una correspondència unívoca entre el conjunt de termes falsos i la *query* per tal d'afegir-los sempre en el mateix lloc. Així no es poden deduir els termes originals si es fa la mateixa *query* més d'un cop.

TrackMeNot (Howe i Nissenbaum, 2011): a diferència de GooPIR, que afegeix soroll a nivell de *query*, TrackMeNot afegeix soroll a nivell de sessió. Per aquesta raó no es generen paraules, sinó *queries* senceres que s'envien de manera aleatòria en el temps. Per tal de fer les cerques més creïbles i dificultar la seva detecció, es generen mitjançant textos de diferents *feeds* RSS de notícies d'actualitat. Té un desavantatge clar respecte del GooPIR i és que el seu ús sobrecarrega la xarxa.

UPIR (Domingo-Ferrer *et al.*, 2009): aquest sistema es basa en una xarxa *peer-to-peer* d'usuaris que s'envien les *queries* a través de canals criptogràficament segurs i realitzen les cerques uns en nom d'altres. Observem que aquest protocol, malgrat el seu nom, formalment no és un protocol PIR. Efectivament, segueix l'estratègia dels mescladors i no emmascara el contingut de les *queries*, sinó la identitat de l'usuari que les ha enviades.

No obstant, tots aquests protocols obstrueixen la personalització que es pot fer a partir dels registres de *queries* pel fet que en redueixen la seva utilitat: GooPIR i TrackMeNot hi afegeixen soroll i UPIR barreja *queries* de diferents usuaris en un mateix *log*.

Finalment, val a dir que també hi ha un treball en compartició de secrets on es proposa un enfocament del problema semblant al d'aquest projecte (Adar, 2007). En aquest article es presenta un mecanisme per detectar *queries* que comprometen la privadesa de l'usuari. Tot i així, no es té en compte l'ús posterior dels seus registres. Es tracta, més aviat, d'un estudi per caracteritzar aquest tipus de cerques que no pas d'un mètode que pugui ser dut a la pràctica.

1.2 El compromís entre Privadesa i Personalització

Així doncs, constatem que no existeix cap protocol de recuperació d'informació de la Web que consideri l'ús que es fa de les dades (*profiling*) i que, a més, pugui ser implementat sense suposar cooperació dels cercadors.

El principal inconvenient que té el disseny d'un protocol d'aquestes característiques és el compromís entre la privadesa i la utilitat de les dades: com més informació s'ofereix, més bona personalització obtenim però, a la vegada, més fem perillar la nostra privadesa. Com que els *logs* del servidor seran analitzats mitjançant tècniques de mineria de dades[8] (*data mining*) per tal de crear els perfils, la millor solució és trobar un compromís entre la quantitat d'informació que es revela i la privadesa de l'usuari. La dificultat en mesurar quanta privadesa o quanta personalització s'està intercanviant recau en la vaguetat d'aquests dos conceptes. Per una banda, no es pot mesurar la utilitat dels registres sense saber com es faran servir per a personalitzar i, òbviament, les companyies dels motors de cerca mantenen en secret les tècniques de personalització que utilitzen. D'altra banda, el grau de privadesa de les dades és completament subjectiu. Per algunes dades concretes com els números de targetes de crèdit és evident, però per altres no és tant clar. Per exemple, un usuari pot no tenir cap inconvenient en fer pública la seva data de naixement, però un altre pot preferir publicar només el dia i el mes del seu aniversari i un tercer, pot voler ocultar-la completament. A continuació definirem el punt de vista que donem en aquest treball sobre aquests dos conceptes.

1.2.1 Definint la privadesa

L'enfocament que donarem sobre la privadesa és independent del judici de l'usuari. Es basa en evitar tècniques de mineria de dades que permetin extreure nova informació de l'usuari, o sigui, es centra en reduir el Risc de Revelació de la seva informació (en anglès, *Disclosure Risk*).

Es considera que hi ha dos tipus de revelació d'informació:

Identity Disclosure: es refereix a enllaçar el registre de dades amb una identitat real (*reidentificació*). Un exemple d'aquest primer tipus és el que van dur a terme els periodistes del NYT per a identificar la Sra. Arnold a partir de les seves cerques.

Attribute Disclosure: en aquest cas la informació no és suficient per identificar l'usuari però ens dóna nova informació sobre ell. En el context dels cercadors, imaginem

[8]Disciplina que consisteix en extreure informació no trivial de les dades que hi resideix implícita.

que un usuari fa cerques sobre una malaltia en concret. Llavors, podem suposar amb una certa probabilitat que l'usuari o alguna persona propera a ell la pateix.

1.2.2 Definint la personalització

Normalment la personalització es defineix com el procés d'oferir la informació adequada a l'usuari adequat i en el moment adequat. En el cas particular de Google, segons explica el seu Director de Gestió de la Producció Jack Menzel, aquesta personalització es duu a terme de dues maneres diferents (Menzel, 2011). La primera utilitza dades de la connexió per localitzar geogràficament a l'usuari (a través de l'adreça IP) i detectar l'idioma, el domini web, etc. Així es poden seleccionar les webs que, per proximitat, li poden ser més útils. També es té en compte la data de la consulta i els esdeveniments propers a ella. Per exemple, cerques sobre focs artificials donen diferents resultats la vigília de Sant Joan que en qualsevol altre moment de l'any. L'altre tipus de personalització utilitza el text de les *queries* i el *data click-through* [9] per crear un perfil de l'usuari.

En aquest treball considerarem només el segon tipus i, en particular, només tindrem en compte la personalització que utilitza els registres de *queries*. Per desgràcia, no la podem avaluar numèricament perquè, com hem dit, no coneixem quins són els algorismes de personalització que fa servir el cercador. Per tant suposarem que, donada una *query*, un registre és útil per a personalitzar la llista de resultats si la interpretació de l'usuari de la *query* té una relació semàntica amb la resta de *queries* del registre.

Donem un exemple per aclarir aquesta definició: suposem que un usuari envia la *query* "*apple*". Aquesta cerca és ambigua perquè tant es pot referir a una fruita com a una companyia. Si el seu registre del servidor conté *queries* que evidencien una predisposició de l'usuari cap a temes d'informàtica, els algorismes de personalització poden reordenar els resultats prioritzant els de la companyia.

1.3 Objectius i Fonaments

L'objectiu general d'aquest projecte és l'estudi i desenvolupament d'un agent[10] per la recuperació d'informació privada a través d'un motor de cerca.

Aquest agent ha de reduir el risc de revelació d'informació de l'usuari i, a més, assegurar una certa personalització de la cerca d'acord amb les definicions de la secció anterior. Ara

[9] Resultats de la cerca que l'usuari va clicar.
[10] Des del punt de vista d'un programa que s'executi i prengui decisions sense necessitat d'interacció per part de l'usuari.

bé, per aconseguir-ho no podrem suposar cap cooperació per part dels motors de cerca, o sigui, s'executarà en l'ordinador de l'usuari i actuarà com un *proxy* entre ell i el cercador.

La hipòtesi principal del projecte és que els usuaris són individus polifacètics, és a dir, no només cerquen sobre un tema en concret sinó que estan interessats en àrees temàtiques molt diverses. Conseqüentment, creiem que és la unió d'aquests interessos el que conforma la seva identitat i els fa únics. Per tal de protegir aquesta identitat, l'estratègia desenvolupada es basa a dissociar les seves facetes. Per això, l'agent haurà de comptar amb un mecanisme per detectar quins són els interessos de l'usuari i classificar les cerques segons aquests. Aleshores, els conjunts de *queries* definits per aquestes facetes s'hauran d'enviar en registres diferents del servidor, de manera que la identitat de l'usuari quedi dividida i sigui més difícil fer una *reidentificació*.

Per tal de crear aquests nous registres en el servidor, l'agent generarà un conjunt de noves identitats per a l'usuari. Les anomenarem identitats "virtuals" perquè, en realitat, representen les diferents facetes de la persona. Seguidament, mitjançant una adequada gestió de les connexions HTTP crearà una nova sessió en el servidor per cada identitat virtual. Quan l'usuari introdueixi una consulta, la query es classificarà en una faceta i, depenent del resultat, s'enviarà al cercador sota una identitat virtual o una altra. Així, el servidor interpretarà que queries enviades utilitzant identitats virtuals diferents han estat enviades per usuaris diferents i les registrarà en els seus *logs* corresponents.

A causa d'aquest funcionament dissociatiu hem anomenat al futur agent *Dissociating Privacy Agent* (DisPA, d'ara endavant).

A continuació desglossem els objectius principals del projecte:

- Dissenyar un model per solucionar el problema del compromís entre privadesa i personalització.
- Implementar el model en un agent com un *add-on*[11] per a l'explorador.
- Desenvolupar un *framework* per avaluar l'agent que sigui una base per a futures investigacions.
- Tant l'agent com el *framework* han de ser el més escalables i portables possible.

1.4 Viabilitat

En aquesta secció del treball es detalla la viabilitat del projecte, des d'un punt de vista econòmic, tècnic i legal. Cal dir que la recerca inclosa en aquest projecte final de

[11] *Plug-in*, extensió especialitzada per a un software concret.

carrera s'ha desenvolupat en el marc del projecte d'investigació ARES CONSOLIDER INGENIO[12] i s'ha dut a terme a les instal·lacions de l'IIIA-CSIC[13].

- Viabilitat tècnica. Com hem pogut observar durant la introducció, la cerca privada a la Web ha despertat un gran interès en els darrers anys, tant en la indústria com en la comunitat acadèmica. D'una banda, les investigacions en aquest camp són relativament recents i encara es poden fer moltes aportacions. D'altra banda, el material necessari per portar a terme aquest projecte és només un equip informàtic, del qual ja es disposa. En tot cas, també es compta amb un ordinador d'alta capacitat de còmput proporcionat per l'IIIA-CSIC per si fes falta a l'hora d'executar els experiments.

  Per aquesta raó el projecte és tècnicament viable.

- Viabilitat econòmica. Les despeses del projecte no són més que el consum energètic de les estacions on executarem els experiments i el material d'impressió. L'accés a articles d'investigació ens el proporciona tant la UAB com l'IIIA-CSIC i, a més a més, a Internet podem trobar una gran quantitat d'articles d'accés obert sobre el tema que tractem i que podem descarregar de forma gratuïta. Els conjunts de dades utilitzats són obtinguts via voluntaris que ens deixaran enregistrar les seves cerques. També utilitzarem els conjunts de cerques que AOL va penjar a Internet i que encara poden ser descarregats (Sadetsky, 2006).

  Per aquestes raons el projecte és econòmicament viable.

- Viabilitat legal. Des del punt de vista legal, no fem pública cap dada personal que pugui violar la Llei Orgànica de Protecció de Dades (LOPD). El *dataset* d'AOL té llicència per a ser utilitzat amb finalitat no comercial i els enregistraments de *queries* fetes amb *plugins* de l'explorador es fan amb el consentiment dels usuaris. A més, només fem públics els resultats de la investigació i en cap cas publiquem dades que puguin comprometre la privadesa dels figurants.

  Per aquests motius el projecte és legalment viable.

- Viabilitat general. Un cop analitzats els aspectes tècnics, econòmics i legals del projecte podem concloure que aquest projecte és viable.

[12] Advanced Research on Information Security and Privacy: http://crises-deim.urv.cat/ares/
[13] Institut d'Investigació en Intel·ligència Artificial - Consejo Superior de Investigaciones Científicas: http://www.iiia.csic.es

1.5 Planificació

El projecte està dividit en diferents etapes que detallarem tot seguit.

1. Documentació. Fer una lectura de la literatura i informar-se sobre les estratègies i mètodes existents per la cerca privada a la Web. L'objectiu és fer-se una idea general de l'estat de l'art i tenir recursos a l'abast per solucionar les dificultats que vagin sorgint.

2. Anàlisi del problema. Estudiar les característiques del problema i dissenyar una estratègia pròpia que el solucioni. També caldrà pensar en mesures per avaluar la solució proposada i que s'implementaran en el *framework*.

3. Implementació. Programar l'agent i l'estratègia dissenyades en l'etapa anterior. Dins d'aquesta etapa s'inclouen l'elecció del llenguatge, l'elaboració de diagrames de classes (UML), testeig, etc.

4. Experimentació. Donar format als conjunts de dades. Seguidament, dissenyar i executar els experiments que avaluaran l'agent.

5. Anàlisi dels resultats. Estudiar els resultat obtinguts en l'etapa anterior i extreure'n unes conclusions.

6. Realització de la memòria.

7. Preparació de la defensa del projecte.

A continuació mostrem un diagrama de Gantt del projecte:

WBS	Name	Work
1	Documentació	22d
2	Anàlisi del problema	31d
3	Implementació	65d
4	Experimentació	10d
5	Anàlisi de resultats	16d
6	Realització de la memòria	27d
7	Preparació de la defensa	10d

Figura 1.1: Tasques del projecte.

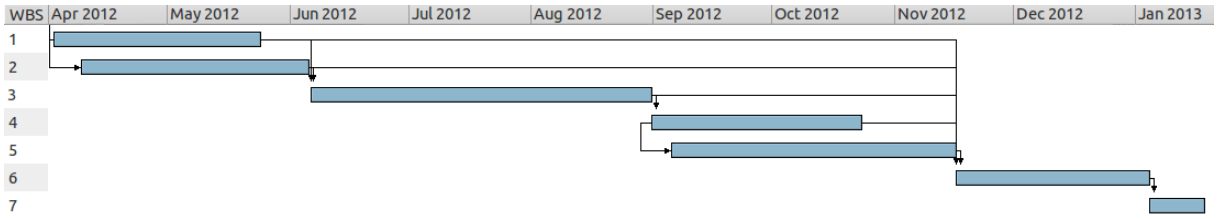


Figura 1.2: Diagrama de Gantt.

1.6 Organització del document

Aquesta memòria s'estructura de la següent manera. En el capítol 2 es dóna un marc teòric per fer aquest treball el més autocontingut possible. En el tercer capítol, es fa l'anàlisi del problema i es presenta el model matemàtic de l'agent. En el capítol 4 s'explica la implementació del mètode descrit en el capítol 3. Seguidament, en el capítol 5 es parla del desenvolupament del *framework* d'avaluació i l'*add-on* pel Firefox. A continuació, en el capítol 6 es dissenyen els experiments i es donen els resultats. Per últim, en el capítol 7 es donen unes conclusions extretes dels resultats i algunes línies de treball futur. Dos capítols més conclouen el projecte, un capítol per als annexes i un per a la bibliografia consultada.

Capítol 2

Marc teòric

En aquest capítol introduirem alguns conceptes previs que són fonamentals per a l'estratègia que segueix l'agent i que serà explicada en detall en el capítol 3.

2.1 El problema de classificació de *queries*

Com ja hem introduït en el capítol anterior, una de les parts crítiques de l'agent serà la Classificació de Queries (QC, per abreujar) en categories semàntiques que descriguin les facetes de l'usuari. El registre del servidor on l'agent les registrarà variarà depenent del resultat d'aquesta classificació.

La classificació de *queries* resulta ser un problema molt més complex que la classificació d'altres tipus de text a causa d'una sèrie de propietats que les caracterítzen i que llistem tot seguit.

(i) Són curtes i disperses. Per exemple: "*ff*", "*ps*".

(ii) No tenen estructura sintàctica. Molts cops els usuaris es limiten a escriure paraules clau ja que les paraules buides de significat (conjuncions, articles, etc.)[1] són eliminades pel propi cercador. Un exemple pot ser: "*play game computer*".

(iii) Contenen soroll. Per exemple, errors tipogràfics: "*facebok*", "*windowd*".

(iv) Apareixen noves paraules contínuament. Un exemple recent: "*iphone5*".

(v) Són curtes i manquen de context.

[1] Conegudes en anglès com *Stop-Words*.

(vi) Tenen faltes d'ortografia que, a més d'afegir soroll, poden fer difícil el reconeixement d'entitats amb nom (*Named Entities*). Per exemple, si s'ometen les majúscules dels noms propis.

Aquestes característiques, juntament amb el caire polisèmic de les paraules, fan que les *queries* siguin ambigües i en alt grau, interpretables. Per exemple, tal i com hem vist que passava amb la *query* "*apple*". Per tant, la seva classificació és subjectiva i depèn de l'usuari. A més a més, la falta de conjunts de dades de *queries* reals fa que sigui difícil utilitzar tècniques d'aprenentatge automàtic per entrenar un classificador. D'una banda, això és pel fet que els proveïdors de cerques no volen caure en el mateix error que AOL. D'altra banda, els usuaris són reticents a etiquetar les seves cerques i, encara més, a publicar-les.

Així doncs, el problema general de la QC tracta de desfer aquesta ambigüitat i s'enuncia de la següent manera:

Definició 1. *(Problema QC)*
Considerem una query q i un conjunt de categories $C = \{c_1, \ldots, c_n\} \subseteq \mathcal{C}$, on $\mathcal{C}$ és una taxonomia de categories. El problema de Classificació de Queries tracta de determinar la rellevància de cada categoria c_i per a la query q.

L'any 2005, l'ACM KKD Cup[2] va tractar aquest problema. Al voltant d'uns 40 equips de tot el món van competir per classificar un conjunt de 800 *queries*. Aquestes consultes havien estat etiquetades per tres persones diferents utilitzant una taxonomia donada per la pròpia organització. D'ençà d'aleshores, la QC ha rebut més i més atenció per part de la comunitat acadèmica i de la indústria.

Una de les dificultats més importants del repte era la poca quantitat de *queries* d'exemple que s'oferien als participants, no suficients per entrenar un classificador. Per això, la majoria dels treballs que es van presentar es centraven a utilitzar recursos de la Web per obtenir dades addicionals i construir models de classificació. Per exemple, la solució guanyadora (Shen *et al.*, 2005) va resoldre el problema de la falta de dades descarregant documents de l'Open Directory Project (ODP). L'ODP, també anomenat *dmoz*, és el directori web més gran editat i mantingut per éssers humans. Els seus editors classifiquen recursos web en una taxonomia que ha sorgit de manera natural, afegint-hi una descripció i unes paraules clau. L'ús de *crawlers*[3] de la Web per a aquesta tasca encara no dóna bons resultats i, per això, l'ODP és una de les classificacions més fiables.

[2]La KDD Cup és la competició anual en mineria de dades i aprenentatge automàtic (*Data Mining and Knowledge Discovering*) que organitza l'ACM (*Association for Computing Machinery*). El 2005 es titulava "*Internet user search query categorization*".
[3]Programes que salten d'una web a una altra per a recollir informació.

Els guanyadors del concurs, primer van definir una correspondència entre les categories de l'ODP i les de la taxonomia objectiu, proposada per la organització. Aquesta correspondència es basava en les paraules clau de les categories. Així, si una categoria c_1 contenia paraules clau en comú amb una categoria de la taxonomia del concurs c_2, es podia aplicar c_1 a c_2. Després introduïen cada *query* en un cercador web i, analitzant les paraules dels *snippets*[4], obtenien un vector de N atributs que servia d'entrada per a una Màquina de Vector de Suport (*Support Vector Machine*, SVM). També van utilitzar els documents descarregats de l'ODP per a generar noves *queries* d'exemple per al classificador. Cal dir que en articles posteriors van perfeccionar la correspondència entre les dues taxonomies (Shen *et al.*, 2006).

Altres recerques han seguit l'estratègia d'utilitzar l'ODP i han obtingut bons resultats. Existeix un treball interessant on es canvia el punt de vista del problema, que fins ara s'havia abordat des de l'aprenentatge automàtic, i es tracta com un problema de Recuperació de la Informació (Ullegaddi i Varma, 2010) . En aquest article utilitzen l'ODP per a construir un motor de cerca propi. Els documents de l'ODP són indexats amb l'objectiu de fer un tipus especial de cerca anomenat *Faceted Search*. Consisteix a indexar els documents per dos camps: un per a la categoria de la taxonomia on es troba el document i un altre pel contingut del document. Aquest tipus de cerca no retorna com a resultat un conjunt de documents, en canvi, retorna un conjunt de categories i el número de *hits*[5] per a cadascuna. Així, la rellevància d'una categoria per a una *query* s'aproxima pel número de *hits* que ha obtingut en la cerca.

En aquesta recerca hem adoptat aquest últim mètode perquè els experiments demostren que millora els resultats de la solució guanyadora de la KDD Cup, és simple i, a més a més, es pot aplicar en un escenari com el nostre, on la classificació ha de fer-se intercalant-se amb les connexions HTTP de la manera més ràpida possible. Tanmateix, considerem que no podem fer servir les tècniques d'extensió i enriquiment de les *queries* que es basen a utilitzar altres cercadors. L'agent ha de tenir un funcionament que eviti donar el mínim d'informació de l'usuari al servidors d'Internet i aquest tipus de tècniques entra en contradicció amb aquest principi. En el seu lloc es poden aplicar estratègies que utilitzen WordNet (Montoyo *et al.*, 2001). WordNet és una ontologia de la llengua anglesa on els noms són organitzats en conjunts de sinònims anomenats *synsets*. Hi ha establertes relacions entre aquests conjunts com, per exemple, la hiperonímia i la hiponímia. Una extensió com aquesta consistiria a trobar termes sinònims, hiperònims i hipònims per

[4]Petit resum del document que ha rebut un *hit* agafant parts del text del document on es troben les paraules de la cerca.
[5]Encerts dels termes de la *query* en un document.

afegir-los a la cerca i, així, augmentar la probabilitat de *hit* de la *query* sobre l'índex.

2.2 Personalització de la cerca

El procés de personalització es porta a terme recollint informació de l'usuari, analitzant-la i desant-la en un perfil. Aquesta, pot ser capturada de dues formes: explícitament, per exemple, demanant un *feedback* mitjançant avaluacions que fa l'usuari manualment; o implícitament, observant el comportament de l'usuari, per exemple, mesurant el temps que dedica a llegir cada document. La captura explícita té un problema important. Si es fa de manera contínua pot resultar una molèstia i, si es fa un sol cop, com que les preferències de l'usuari canvien amb el temps, el perfil que s'haurà creat acabarà sent incorrecte. Per aquesta raó hi ha tants treballs realitzats utilitzant la personalització implícita (Billsus i Pazzani, 1999; Liu *et al.*, 2002; Chen *et al.*, 2002).

Per fer una bona captura implícita s'han de tenir en compte els interessos tant a curt com a llarg termini. A més, s'ha d'incloure un model del context de la cerca que permeti detectar els llindars entre sessions, on les *queries* són generades per aconseguir un mateix objectiu. Per exemple, quan anem reformulant la consulta per mitjà d'afegir nous termes (*Vertical Search*). Durant aquest procés introduïm múltiples *queries* però totes comparteixen la mateixa intenció.

La personalització des dels servidors normalment s'aconsegueix mitjançant l'anàlisi de:

(1) L'historial de *queries*.

(2) Els enllaços dels resultats on ha fet clic.

En el punt (2), la idea és que els enllaços que ha clicat ens ajuden a determinar la interpretació que dóna a la cerca. Per exemple, existeix un model de classificació col·laboratiu on es suposa que usuaris que coincideixen en els clics dels resultats per la mateixa query, tenen interpretacions similars d'aquesta.

Tenint en compte aquests punts, hem de fer un parell de consideracions respecte al nostre agent. Per un costat, creiem que deixar que el cercador capturi quins enllaços ha clicat és cedir massa informació. Per aquesta raó, descarregarem els enllaços dels resultats a disc i els mostrarem de manera local. Per tant, el cercador no podrà personalitzar a partir de (2). Per altre costat, una personalització implícita a través de l'agent s'escapa de l'abast d'aquest projecte. Per això, en les seccions que venen a continuació, només ens dedicarem a *mantenir* la personalització que es pot fer a partir de (1) (recordem la definició de personalització donada en la Introducció).

Capítol 3

Model

En aquest capítol descriurem un model matemàtic per tal de resoldre el problema del compromís entre personalització i privadesa associat a les cerques Web. Representarem el servidor d'un cercador com una tupla $S = (Q, U)$, on U és el conjunt de tots els usuaris del servidor i Q el conjunt de totes les *queries* (sense considerar l'ordre dels termes de les *queries* i agafant totes les *queries* repetides com una de sola). Denotarem per $Q(u) \subseteq Q$ el conjunt de *queries* que està associat a un usuari $u \in U$. Aquest conjunt representarà el registre de l'usuari al servidor.

3.1 Caracteritzant el Risc de Revelació d'Informació

En primer lloc, definirem mesures que avaluïn el Risc de Revelació d'Informació que pateix un usuari. Sigui $q \in Q$ una *query*, el conjunt

$$U(q) := \{u \in U \mid q \in Q(u)\}$$

és el conjunt d'usuaris que tenen la *query* q registrada en el seu registre del servidor. El cardinal d'aquest conjunt mesura la *freqüència de la query* al servidor.

Definició 2. *(Freqüència d'una query)*
Sigui $S = (Q, U)$ un servidor i $q \in Q$ una query, la freqüència de q és:

$$f(q) := |U(q)| .$$

Observem que, donat un usuari $u \in U$, la intersecció

$$\bigcap_{q \in Q(u)} U(q)$$

ens dóna informació sobre l'anonimitat de l'usuari dins del servidor. Per exemple, si una combinació de *queries* ha estat cercada només per u, el registre de l'usuari és únic dins del servidor. Si, en canvi, hi ha $k > 1$ usuaris dins d'aquesta intersecció, els seus logs seran indistingibles. No obstant, aconseguir aquesta segona opció és pràcticament impossible en servidors reals a causa de la dificultat de trobar dos usuaris que tinguin els seus registres de *queries* idèntics. Un requeriment per aconseguir-ho és enviar sempre *queries* amb freqüència major que 1. És una restricció difícil d'acceptar pels usuaris i, tot i així, tampoc podríem assegurar la k-anonimitat[1]. Per això proposem una relaxació d'aquest concepte.

Definició 3. *(Anonimitat de l'usuari dins del servidor)*
Sigui $S = (Q, U)$ un servidor. Considerem un usuari $u \in U$. L'anonimitat de l'usuari és:

$$A(u) := \min_{q \in Q(u)} \{f(q)\}.$$

Així doncs, l'anonimitat de l'usuari és la mínima de les freqüències de totes les *queries* del registre de l'usuari.

Una segona mesura que està relacionada amb el risc de revelació d'atributs (*Attribute Disclosure*) és la distància mitjana del registre.

Definició 4. *(Distància mitjana normalitzada del registre)*
Sigui $S = (Q, U)$ un servidor i sigui $d : Q \times Q \longrightarrow \mathbb{R}^+$ una funció de semblança entre queries afitada (una exemple pot ser la distància entre paraules definida amb l'ús de diccionaris que es descriu a Torra *(2007)). Donat un usuari $u \in U$, la distància mitjana normalitzada del registre de u és*

$$D(u) := \frac{1}{\mu \ |Q(u)|} \sum_{q \in Q(u)} d(q_c, q),$$

on q_c és un centroid del registre (a la pràctica podem agafar una query del log a l'atzar) i μ és la distància màxima que es pot mesurar, per tal de normalitzar la distància mitjana en l'interval $[0, 1]$.

[1] En el sentit que hi ha k registres d'usuari indistingibles.

Atès que la hipòtesi fonamental del projecte és que la diferència entre les *queries* està definida per les facetes de l'usuari, la funció d es basarà en la semblança dels significats de les paraules[2] (un contraexemple podria ser una mesura de la distància textual comparant diferències dels símbols, e.g. *plat* és més proper a *platan* que no a *cullera*). Així, observem que si les *queries* són molt semblants entre elles (distància mitjana baixa), donen poca nova informació. En canvi, com més diferents siguin entre elles (distància mitjana alta), més quantitat d'informació podem extreure combinant-les.

Definició 5. *(Índex de Disclosure Risk)*
Sigui $S = (Q, U)$ un servidor. Considerem un usuari $u \in U$. L'Índex de Disclosure Risk de u és

$$IDR(u) := \frac{1}{A(u)^{1-D(u)}}$$

Aquest valor sempre es troba en l'interval $[0, 1]$ i expressa el Risc de Revelació d'Informació segons el nostre model. Si fixem l'anonimitat i ens mirem aquesta expressió respecte la distància mitjana, el risc augmentarà a mesura que la distància mitjana augmenti. Si és zero, per exemple, si el registre conté una sola *query*, el risc només dependrà de l'anonimitat. Observem que llavors el risc es dividirà pel número d'usuaris que la tinguin en el seu registre però mai arribarà a ser zero. Per exemple, si tenim $A(u) = k$, aleshores totes les *queries* hauran estat fetes per k usuaris diferents i, per tant, el risc es dividirà per k. En cas de tenir distància mitjana màxima, com quan tenim *queries* el més distants entre elles possible, el risc sempre serà 1. La raó és que, per molt que tinguem molta anonimitat i les *queries* hagin estat repetides per altres usuaris, la unió de queries molt diferents suposa un risc màxim del *Attribute Disclosure*.

Per últim, definirem l'Índex de Disclosure Risk d'un conjunt d'usuaris com el risc de l'usuari que té màxim risc de revelació.

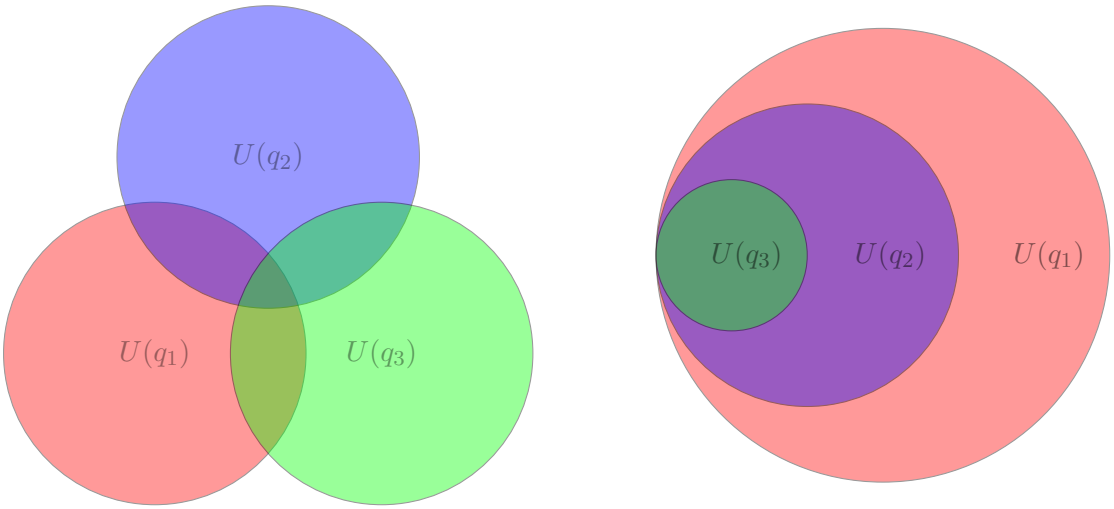
Definició 6. *(Índex de Disclosure Risk conjunt)*
Sigui $S = (Q, U)$ un servidor. L'Índex de Disclosure Risk de $u_1, \ldots, u_n$ és

$$IDR(u_1, \ldots, u_n) := \max_{i=1,\ldots,n} \{IDR(u_i)\}$$

Vegem com el nostre model descriu correctament la realitat. En la Figura 3.1 observem dues possibles distribucions dels conjunts $U(q)$ per a tres *queries* diferents. En el cas 3.1a hi ha pocs usuaris que les tinguin en comú perquè són molt diferents, per exemple, si pertanyen a facetes diferents de l'usuari. En la intersecció trobaríem usuaris

[2] La distància semàntica entre elements lingüístics és una mesura de la semblança dels seus significats.

polifacètics (cas normal). En aquest cas l'anonimitat és baixa i la distància mitjana és alta, conseqüentment, l'Índex de Revelació d'Informació és molt alt. La distribució del cas 3.1b correspon a usuaris que només estan interessats en un sol tema (cas patològic), de més a menys especialitzats: els usuaris de $U(q_3)$ han cercat les mateixes *queries* que els usuaris de $U(q_2)$ i $U(q_1)$ i, els de $U(q_2)$, les *queries* dels de $U(q_1)$. En aquest segon cas la distància mitjana és baixa i el *disclosure risk* només depèn de l'anonimitat, en concret la freqüència de q_3 en serà una fita.



(a) Distància mitjana elevada.

(b) Baixa distància mitjana.

Figura 3.1: Dues possibles distribucions dels conjunts $U(q_i)$

3.2 Dissociació d'identitat

L'estratègia que definim en aquesta secció s'anomena dissociació d'identitat i veurem com pot reduir el risc de revelació d'informació de u. Es basa en la generació de n nous usuaris $u_1, \ldots, u_n$ tals que

$$Q(u) = Q(u_1) \cup \cdots \cup Q(u_n),$$

on $Q(u_i)$ són no buits i disjunts dos a dos.

A continuació definirem formalment aquests conceptes i després mostrarem com aquest procés pot reduir el risc de revelació.

Definició 7. *(Dissociació d'un usuari u)*
Sigui $S = (Q, U)$ un servidor. Considerem un usuari $u \in U$. Una dissociació de u és una relació d'equivalència $\sim$ definida sobre el seu registre de queries $Q(u)$.

Definició 8. *(Identitats Virtuals)*
Sigui $S = (Q, U)$ un servidor. Considerem un usuari $u \in U$. Si $\sim$ és una dissociació de u, una identitat virtual de u és un usuari $u' \in U$ tal que $Q(u')$ és una classe d'equivalència d'aquesta dissociació.

Per a un usuari u i una dissociació de u denotem les seves identitats virtuals amb subíndexs: $u_1, \ldots, u_n$ i, per a simplificar la notació, el conjunt d'identitats virtuals per:

$$idv(\sim) := \{u_1, \ldots, u_n\}.$$

A continuació demostrarem que una dissociació mai pot augmentar el risc de revelació segons el nostre model.

Proposició 1. *Sigui $S = (Q, U)$ un servidor. Considerem $u \in U$ un usuari amb un registre de queries $Q(u)$. Donada una dissociació $\sim$ sobre $Q(u)$ i $idv(\sim) = \{u_1, \ldots, u_n\}$ les respectives identitats virtuals, aleshores*

$$IDR(u_i) \leq IDR(u), \qquad \forall i = 1, \ldots, n$$

Demostració:

Només hem de veure que $A(u) \leq A(u_i)$ i $D(u) \geq D(u_i)$ per qualsevol $i = 1, \ldots, n$. Per un costat tenim que

$$\begin{aligned}
A(u) &= \min_{q \in Q(u)} \{f(q)\} \\
&= \min_{q \in Q(u_1) \cup \cdots \cup Q(u_n)} \{f(q)\} \\
&= \min \left\{ \min_{q \in Q(u_1)} \{f(q)\}, \ldots, \min_{q \in Q(u_n)} \{f(q)\} \right\} \\
&\leq \min_{q \in Q(u_i)} \{f(q)\} \qquad \forall i = 1, \ldots, n \\
&= A(u_i) \qquad \forall i = 1, \ldots, n
\end{aligned}$$

Per l'altre costat tenim

$$
\begin{aligned}
D(u) &= \frac{1}{\mu\ |Q(u)|} \sum_{q \in Q(u)} d(q_c, q) \\
&= \frac{1}{\mu} \left(\frac{1}{|Q(u_1)|} \sum_{q \in Q(u_1)} d(q_c, q) + \ldots + \frac{1}{|Q(u_n)|} \sum_{q \in Q(u_n)} d(q_c, q) \right) \\
&\geq \frac{1}{\mu\ |Q(u_i)|} \sum_{q \in Q(u_i)} d(q_c, q) \qquad \forall i = 1, \ldots, n \\
&= D(u_i) \qquad \forall i = 1, \ldots, n
\end{aligned}
$$

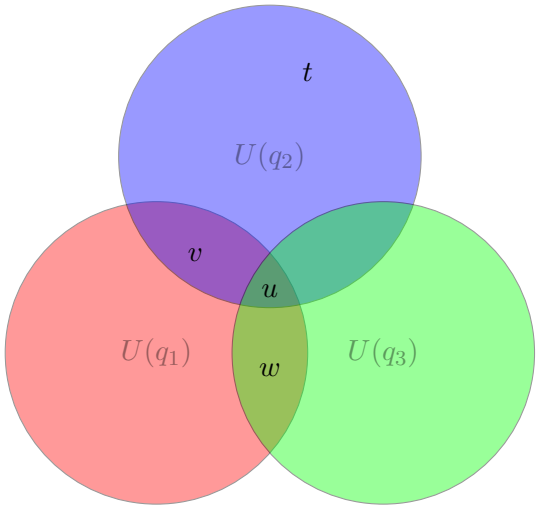
Del fet que $g(x) = a^{1-x}$ *i* $h(x) = \frac{1}{x}^b$ *són creixents en* $[0, 1]$ *per a* a, $b \in [0, 1]$ *constants, es desprèn el resultat de la proposició.* □

Vegem un exemple que il·lustri aquest resultat.

Exemple 1. *Imaginem que tenim els usuaris* u, v, w, $t \in U$ *i les queries* q_1, q_2, $q_3 \in Q$. *A la Figura 3.2 tenim, a l'esquerra els logs del servidor i, a la dreta, el diagrama de Venn dels conjunts* $U(q_i)$.

$Q(u)$	$Q(v)$	$Q(w)$	$Q(t)$
q_1	q_1	q_1	
q_2	q_2		q_2
q_3		q_3	

(a) Logs del servidor.



(b) Diagrama de Venn.

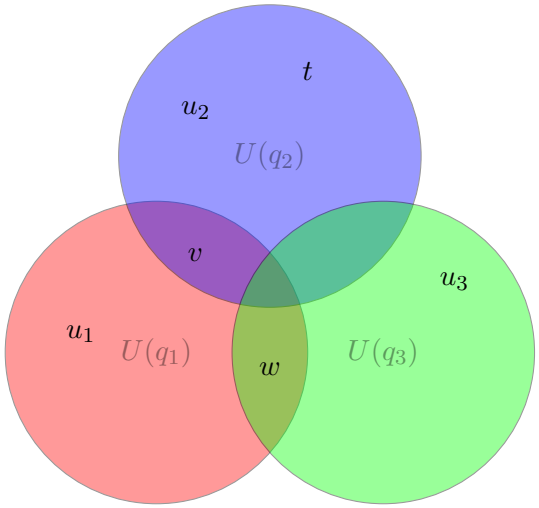
Figura 3.2: Distribució dels conjunts $U(q_i)$

En aquest cas, l'usuari es troba a la intersecció i té un IDR elevat. Aleshores, si

dissociem u segons una relació d'equivalència $\sim$ que ens fa tres particions del conjunt de queries $Q(u)$, tindrem tres identitats virtuals $u_1, u_2, u_3 \in U$ tals que $Q(u_1) = \{q_1\}$, $Q(u_2) = \{q_2\}$ i $Q(u_3) = \{q_3\}$. La distribució en aquest cas es mostra en la Figura 3.3.

u_1	u_2	u_3	v	w	t
q_1			q_1	q_1	
	q_2		q_2		q_2
		q_3		q_3	

(a) Logs del servidor.



(b) Diagrama de Venn.

Figura 3.3: Distribució de $U(q_i)$ després de la dissociació

Ara el risc s'haurà reduït ja que per cadascuna de les identitats virtuals, la distància mitjana ha disminuït (de fet és zero perquè cada registre conté una sola query) i l'anonimitat s'ha mantingut igual (però observem que podria haver augmentat només que haguéssim tingut algun usuari més a $U(q_3)$).

Donada una dissociació, considerarem que el Risc de Revelació d'Informació de l'usuari dissociat és $IDR(idv(\sim))$, és a dir, l'Índex del Risc de Revelació conjunt de les identitats virtuals.

Així, fixat un usuari que volem protegir u, considerem l'aplicació

$$\begin{array}{rccc} G_u: & \mathcal{R}(Q(u)) & \longrightarrow & [0,\,1] \\ & x & \longmapsto & IDR(idv(x)) \end{array}$$

on $\mathcal{R}(Q(u))$ és el conjunt de relacions d'equivalència a $Q(u)$. Aquesta funció avalua el risc de revelació de l'usuari per a cada possible dissociació. Observem que per la proposició anterior, per a cap dissociació aquest risc serà major que el risc inicial de l'usuari. Podem prendre G_u com la nostra funció objectiu a minimitzar. Això sí, hem d'afegir una restricció: el problema d'optimització estarà subjecte a una condició sobre la

utilitat dels registres. Així doncs, sense entrar en detalls sobre aquesta funció d'utilitat, el problema es pot expressar de la manera següent.

$$\begin{aligned} &\text{minimitzar} && G_u(x) \\ &\text{subjecte a} && \mathcal{U}(x) - \mathcal{U}_{min} > 0 \end{aligned}$$

on $\mathcal{U}$ és la funció d'utilitat dels registres de l'usuari per a una dissociació x. $\mathcal{U}_{min}$ és un fita inferior d'aquesta funció. Establim aquesta fita per assegurar un mínim d'utilitat dels registres dissociats.

En les següents seccions explicarem un mètode per solucionar aquest problema d'optimització.

3.3 Mètode de l'agent

Com que la distància mitjana i la utilitat estan relacionades perquè es basen en la semàntica de les *queries*, seguirem la següent estratègia per a minimitzar la funció objectiu G_u. En primer lloc minimitzarem la distància mitjana tenint en compte la utilitat dels registres (passar del cas 3.1a al cas 3.1b) i, atès que, com hem dit, amb distància mitjana mínima el risc només depèn de l'anonimitat, després maximitzarem l'anonimitat.

3.3.1 Minimitzant la distància mitjana

En primer lloc, observem que una solució trivial del problema és la relació unària que relaciona cada *query* només amb ella mateixa:

$$q_i \sim q_j \iff q_i = q_j.$$

En aquesta situació obtindríem una identitat virtual per a cada nova *query*. El problema és que la utilitat dels registres resultants serà pràcticament nul·la. Per a trobar un compromís entre distància mitjana i utilitat proposem la següent dissociació.

Definició 9. *(Dissociació DisPA) Sigui $u \in U$ un usuari del servidor i $C(u) = \{c_1, \ldots, c_2\}$ un conjunt de categories*[3]*. Suposem que podem classificar tota $q \in Q(u)$ en una categoria de $C(u)$. Siguin $q_1, q_2 \in Q$, tals que $classif(q_1) = c_1$ i $classif(q_2) = c_2$. Aleshores,*

$$q_1 \sim q_2 \iff c_1 = c_2.$$

[3]Tal i com les definíem en el capítol 2 pel problema de classificació.

Tal i com hem comentat a la introducció, el conjunt de categories $C(u)$ descriurà les facetes de l'usuari, i.e., les principals àrees d'interès d'aquest. Suposant que una *query* només va a parar a una categoria, és trivial veure que la relació de l'anterior definició és d'equivalència. A més a més, observem que els registres del servidor resultants compliran la propietat que enunciàvem a la introducció quan definíem la utilitat de les dades: les *queries* de $Q(u_i)$ tindran una relació semàntica segons la interpretació de l'usuari. Per altra banda, la distància mitjana també es reduirà perquè les *queries* dels registres finals són properes semànticament.

Posem un exemple pràctic i simple d'aquesta dissociació.

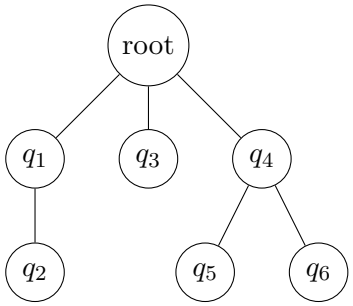
Exemple 2. *En els exemples de la secció anterior no fèiem explícits els termes de les queries perquè encara no consideràvem la seva semàntica. Ara imaginem que volem protegir un usuari u amb el registre de queries representat en la Figura 3.4.*

q_1 = "*listen music*" q_4 = "*futbol match*"

q_2 = "*rock hits*" q_5 = "*barça*"

q_3 = "*apple*" q_6 = "*r madrid*"

(a) Termes de les *queries* de $Q(u)$.



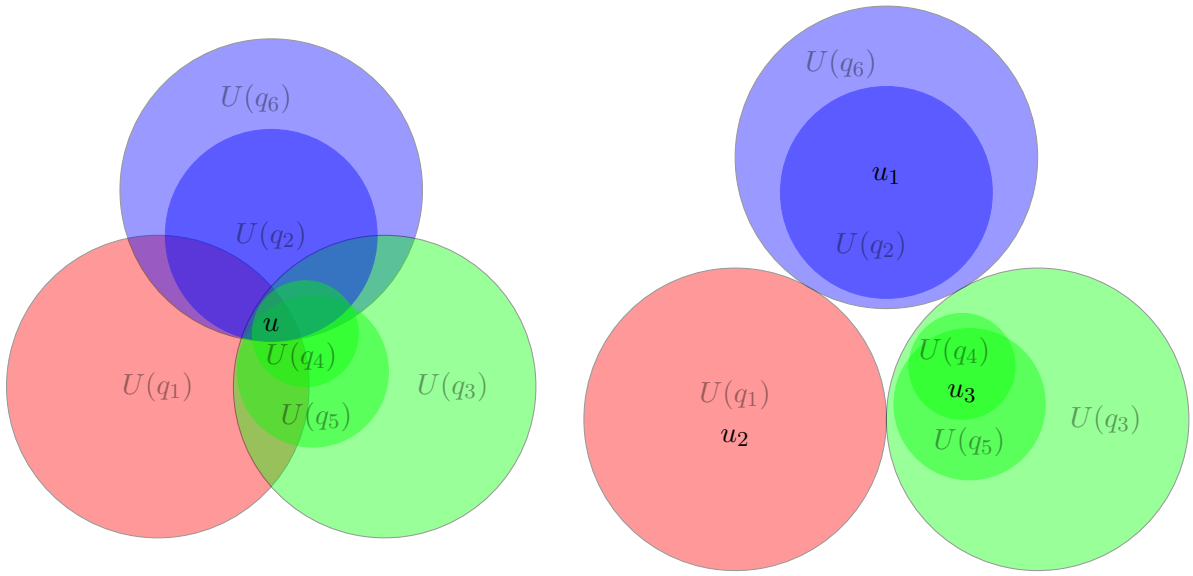
(b) Relació jeràrquica de les *queries*

Figura 3.4: Conjunt de *queries* del servidor $Q(u)$.

Observació 1. *L'estructura jeràrquica de la Figura 3.4b depèn de la interpretació que l'usuari dóna a les queries.*

Llavors dissociem u segons la dissociació DisPA. Suposarem que l'usuari només té tres facetes $C(u) = \{c_1, c_2, c_3\}$, amb c_1 = "Art", c_2 = "Esport", c_3 = "Informàtica". Llavors, per dissociar u crearem tres identitats, una per a cada faceta, que ens identificaran les particions del registre $Q(u)$. Llavors, la distribució quedarà tal i com es mostra en la

Figura 3.5.



(a) Abans de dissociar.

(b) Després de dissociar.

Figura 3.5: Diagrames de Venn d'abans i després de la dissociació semàntica de u.

Tal i com hem vist a l'exemple, el problema de minimitzar la distància mitjana es redueix a inferir les facetes $C(u)$ i a classificar-hi les *queries*. Ja hem introduït les dificultats d'aquest procés en el capítol 2. Un exemple ja familiar és el de la *query* "*apple*", a l'exemple anterior tant podria haver anat dins la categoria d'"Informàtica" com dins d'una categoria anomenada "Fruita".

3.3.2 Model probabilístic de classificació personalitzada

Com hem fet notar a l'observació 1, per tal de conservar la utilitat dels logs del servidor, el classificador de *queries* que fem servir ha de tenir en compte la interpretació de l'usuari. Pensem que si amb "*apple*" l'usuari es refereix a la fruita i classifiquem la *query* dins del log d'informàtica, el cercador extraurà de les *queries* la preferència:

$$\text{"Informàtica"} \prec \text{"Fruita"},$$

és a dir, donarà preferència a informàtica en les *queries* futures.

Com hem dit a la secció 2.2, no és l'objectiu d'aquest treball dotar a l'agent d'un sistema de personalització implícita complet i fer un tractament dels resultats per tal que en retorni els més adequats per a l'usuari. El que volem és facilitar la feina al cercador per

tal que personalitzi el millor possible a partir de les *queries*. L'avantatge és que aquesta personalització serà distribuïda i no s'afegirà còmput extra al cercador.

L'enfocament que donem per a resoldre aquest problema és semblant al de Cao *et al.* (2009). La principal diferència és que nosaltres utilitzarem totes les pàgines webs de l'historial de navegació de l'usuari enlloc de només els enllaços clicats sobre la pàgina de resultats. Això ho podem fer gràcies a que l'agent té accés a molta més informació de l'usuari que no pas un servidor d'un motor de cerca.

En primer lloc, observem que per a una classificació de *queries* simple podem muntar un classificador que maximitzi la probabilitat d'una categoria donada una *query*, és a dir,

$$c = \arg\max_{c_i \in C} p(c_i \mid q).$$

Nogensmenys, l'objectiu és construir un classificador que tingui en compte l'usuari,

$$c = \arg\max_{c_i \in C} p(c_i \mid q, u),$$

on $p(c_i \mid q, u)$ és la probabilitat de la categoria donats un usuari i una *query*.

Per aproximar aquesta probabilitat, utilitzarem un model probabilístic. Aquest, descriu el procés de classificació de la següent manera. Els usuaris cerquen informació que pertany a una categoria c i envien *queries* q a l'agent per tal que les introdueixi per ell en el cercador. Abans, però, l'agent classifica les *queries* en una categoria c. A més a més, la informació que cerquen es troba en webs w que també són classificades per l'agent en aquestes categories. En la Figura 3.6 veiem una representació de les variables i les dependències d'aquest model.

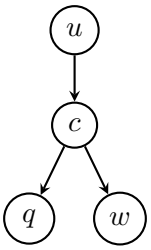


Figura 3.6: Representació gràfica del model probabilístic.

Llavors, podem utilitzar la Fórmula de Bayes per a obtenir la probabilitat a posteriori:

$$p(c \mid q, u) = \frac{p(c, q, u)}{p(q, u)}. \tag{3.1}$$

La probabilitat del numerador és senzilla de calcular segons la xarxa bayesiana del

model:

$$p(c, q, u) = p(u)\, p(c \mid u)\, p(q \mid c).$$

Aleshores, per calcular $p(q \mid c)$ tornem a fer servir la Fórmula de Bayes

$$p(q \mid c) = \frac{p(c \mid q)\ p(q)}{p(c)}. \tag{3.2}$$

i, per calcular $p(c \mid u)$, aproximem de la següent forma:

$$p(c \mid u) \approx p(w_1, \ldots, w_k \mid u), \tag{3.3}$$

on $w_1, \ldots, w_k$ són les k últimes webs que l'usuari ha visitat.

Per tant, podem aproximar $p(c \mid q, u)$ substituint les expressions 3.3 i 3.2 a l'equació 3.1. Simplificant termes obtenim:

$$p(c \mid q, u) \approx \frac{p(q)}{p(q, u)}\, p(c \mid q)\, p(w_1, \ldots, w_t \mid c). \tag{3.4}$$

Finalment, el classificador es pot construir maximitzant aquesta probabilitat respecte les categories:

$$classify(q_0) = \arg\max_{c_i \in C} \left\{ \frac{p(q)}{p(q, u)}\, p(c_i \mid q_0)\, p(w_1, \ldots, w_t \mid c_i) \right\} \tag{3.5}$$

$$= \arg\max_{c_i \in C} \left\{ p(c_i \mid q_0)\, p(w_1, \ldots, w_t \mid c_i) \right\}. \tag{3.6}$$

Observem que, a l'hora de maximitzar, hem obviat tots els termes que no depenien de la categoria perquè eren termes constants.

3.3.3 Maximitzant l'anonimitat

Ara que tenim solucionat el problema de la optimització de la distància mitjana conservant la utilitat, atacarem el problema de maximitzar l'anonimitat.

Maximitzar l'anonimitat significa introduir cerques que hagin estat fetes pel màxim número d'usuaris possible. Com que aquest requisit és molt difícil de complir, començant perquè no coneixem els registres dels altres usuaris, ens conformarem a assegurar un mínim d'anonimitat. Per fer això definirem un llindar per la freqüència de la *query* a partir del qual, enviar-la, implicaria una reducció de l'anonimitat. Anomenarem a

aquesta freqüència f_0. Aleshores, suposant que podem conèixer la freqüència d'una *query*, implementarem un filtre de la següent manera: si $f(q_0) < f_0$, considerarem que la consulta q_0 compromet la privadesa de l'usuari.

A partir d'aquí sorgeixen dos problemes. El primer és que només podem conèixer f_0 si tenim accés als conjunts Q i U del servidor. El segon és determinar l'actuació de l'agent en el cas de detectar una d'aquestes *queries*. En el capítol 4 estudiarem aquests problemes i donarem una aproximació de les freqüències de les *queries* amb la finalitat d'implementar aquest filtre.

Capítol 4

Implementació del mètode

El motor de cerca utilitzat per implementar el mètode és el cercador de Google ja que segons els estudis citats al principi d'aquesta memòria, és el cercador més utilitzat arreu del món. En aquesta secció explicarem com portarem a la pràctica el mètode presentat i, en el capítol 5, com serà desenvolupat en un *add-on* per a l'explorador.

4.1 Mòdul classificador

El classificador s'implementa mitjançant una taxonomia que representem com un arbre on els nodes són categories semàntiques i, les fulles, documents. Fixarem un nivell de l'arbre que representarà les facetes de l'usuari $\mathcal{C} = \{c_1, \ldots, c_n\}$, on c_i són nodes. Els nodes contindran informació sobre classificacions passades. En concret, hi desarem el número de *queries* i el número de llocs web visitats per l'usuari que han estat classificats en aquesta categoria. Llavors, utilitzant un cercador local construit a partir dels documents (fulles de l'arbre), trobarem el número de *hits* de la cerca per a cada categoria, tal i com queda recollit en el capítol 2. Així, les probabilitats del classificador expressades en l'Equació 3.6 s'aproximen en termes d'aquests valors com:

$$classify(q_0) = \arg\max_{c_i \in C} \left\{ h_i(q_0) \, \frac{v_i}{|c_i| + v_i} \right\},$$

on $h_i(q_0)$ és el número de *hits* que ha obtingut la *query* q_0 per a la categoria c_i, v_i el número de llocs web classificats en el node c_i i, $|c_i|$, el cardinal de la categoria, és a dir, el número de fulles que pengen d'aquest node.

Observació 2. *Poden resultar una mica confusos els conceptes d'identitat virtual, categoria i faceta. La principal diferència és que les facetes ja existeixen a priori en la personalitat de l'usuari. En contrast, les categories (nodes de l'arbre) són un model de les*

facetes de l'usuari. En canvi, les identitats virtuals són identificadors d'usuari associats als registres del servidor del cercador, que l'agent crea a propòsit per a fer la dissociació.

La taxonomia escollida ha estat l'Open Directory Project (ODP), ja introduït en el capítol 2. A l'agost de 2012, l'ODP comptava amb 5.066.513 *sites*, 96.160 editors i al voltant de 1.012.859 categories. Tot i que només hi ha indexat el 5% del número total de documents existents a la Web, a causa de ineficiència de les tècniques automàtiques de classificació de webs actuals, és la classificació més acurada que hem trobat. El nivell de la taxonomia que hem fixat ha estat el primer de tots, el qual conté les següents categories: *Adult, Arts, Games, Shopping, Business, Health, Kids and Teens, Society, Computers, Home, News, Reference, Regional, Recreation, Sports, Science, Society, World.* Tanmateix, hem obviat algunes categories que ens han semblat redundants per a la nostra classificació, són: *World*, *Kids and Teens*, *Regional*. Val a dir que la categoria *World*, que conté les mateixes categories que l'arrel però en altres idiomes, també pot ser útil per ampliar l'agent en treballs futurs.

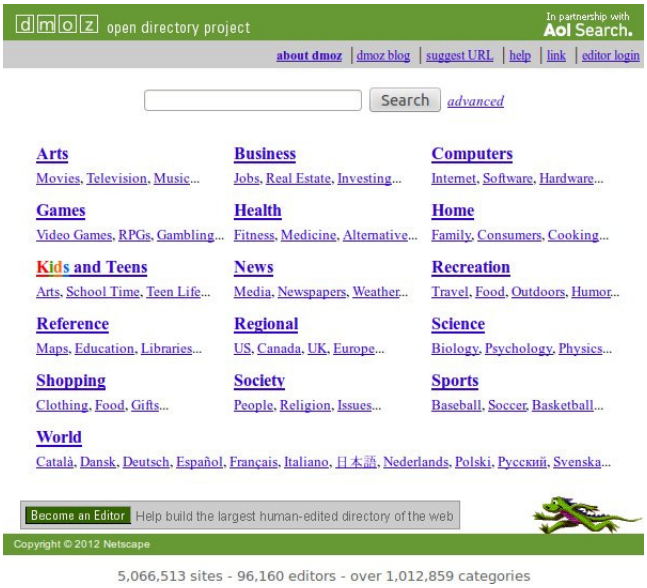


Figura 4.1: Categories de l'ODP

Després de descarregar totes les web indexades, hem fet una neteja de webs que ja no existien. De fet, com que també s'indexen documents en altres formats que no són HTML, el número de documents descarregats ha estat menor de l'esperat. Tot i així, han quedat 2.4 milions de documents en anglès dels 2.6 milions existents. El conjunt de dades final (sense comprimir) ocupa aproximadament 12GB de memòria. Cal tenir present que no només hem descarregat el contingut de la web, sinó les descripcions i paraules clau que els editors introduïen a l'hora d'indexar. També hem *parsejat* les *meta-tags* del codi font

com el *Title*, la *Description* i les *Keywords*. Finalment, hem penjat totes aquestes dades a una base de dades MySQL per tal que fossin més manejables.

El cercador local s'ha implementat amb el framework d'Apache, Lucene. És una llibreria que permet implementar cerques de l'estil *faceted search* fàcilment. També l'hem utilitzat per a crear els índexs a partir dels documents descarregats. L'índex creat és un tipus d'índex anomenat *índex invers*. Es tracta d'una estructura de dades que referencia els documents a partir dels termes del corpus[1]. Així, quan consultem un terme de la *query*, obtenim la freqüència d'aquest terme en cada document. Això permet reduir l'espai de cerca a canvi d'un cost computacional extra a l'hora d'indexar. Per accelerar la classificació, Lucene també permet crear un índex per a la taxonomia. Per últim, amb l'objectiu d'experimentar, hem creat un índex del WordNet, per ampliar les *queries* tal i com explicàvem al capítol 2. Com que els índexs resultants eren molt grans (de l'ordre de GBs), també hem creat altres índexs només utilitzant les paraules clau i obviant els continguts. Així hem aconseguit reduir la seva mida fins a 200MB. Aquests índexs també són més ràpids però la qualitat de la classificació és menor a simple vista. Cal dir que la majoria d'aquests índexs es poden descarregar des de la web del projecte:

http://code.google.com/p/dispa-framework/.

4.2 Mòdul de Gestió dels contextos

A continuació explicarem com es tradueixen els conceptes del model del capítol 3 a l'hora d'implementar l'agent.

En primer lloc, les identitats virtuals són el conjunt d'identificadors que s'inclou en els *logs* del servidor. En la següent secció estudiarem en detall quins són, però observem que els servidors només tenen accés a les dades incloses en el paquet de dades de la connexió.

En segon lloc, la dissociació DisPA és molt semblant a una suplantació d'identitat HTTP (vegeu el capítol 7), on tots els comptes suplantats pertanyen al mateix usuari. A la Figura 4.2 es representa aquest procés de manera simplificada.

La principal estratègia del servidor per a detectar aquest tipus d'atacs consisteix en trobar inconsistències en les dades de la connexió. Per exemple, trobar una adreça incoherent al camp *referral*, camp que indica el recurs web d'on prové la última connexió HTTP. Per tant, per tal de mantenir una sessió consistent i evitar que el servidor detecti l'agent, definim el *Context de la Connexió* com el conjunt de dades de la connexió que

[1]Conjunt de documents a partir del qual es construeix l'índex.

el cercador pot utilitzar per a detectar un comportament maliciós, entre elles la identitat virtual, però també el camp *referral*, totes les *cookies* del domini del servidor, etc.

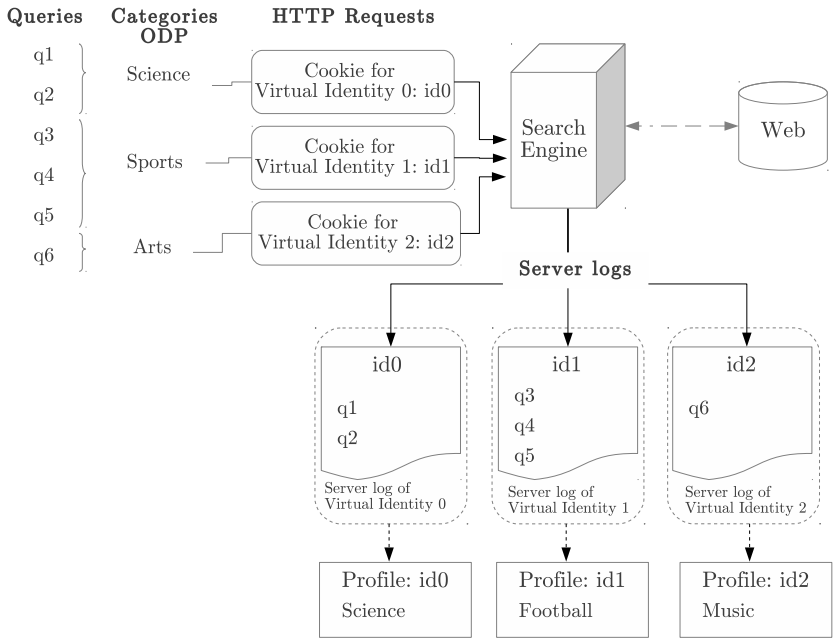


Figura 4.2: Representació de la gestió dels contextos.

Per altra banda, hi ha altres dades que caracteritzen una cerca que associem i que emmagatzemem juntament amb la identitat virtual i el context de la connexió. Definim el *Context* com aquest conjunt de dades. En l'agent DisPA, el context associat a una identitat virtual inclou:

Categoria: categoria de l'ODP.

Context de la cerca: idioma de cerca, subdomini de Google (.cat, .es, .co.uk, etc), data, localització, entre altres.

Context de la connexió: dades relacionades amb la connexió HTTP tal i com hem definit.

Historial de *queries*: *queries* que han acabat en el registre definit per la identitat virtual d'aquest context.

Pàgines de resultats de *queries* anteriors: pàgines de resultats obtingudes en cerques anteriors.

Entitats amb nom: conjunt de noms propis que apareix en totes les *queries* d'aquest context. Més endavant en detallarem el perquè.

Així doncs, l'agent implementarà una gestió d'aquests contexts des de l'explorador. Quan un usuari enviï una consulta, l'agent la classificarà en una certa categoria i, llavors, farà un *bypass* de la comunicació amb el servidor per a canviar el context associat al resultat de la classificació.

En les següents seccions detallarem com es fa aquest procés i com farem el filtre de freqüència.

4.3 Bypass de la connexió

La majoria de llocs web necessiten recordar l'estat de la connexió per tal de gestionar les sessions i personalitzar els continguts. No obstant, el protocol HTTP és un protocol que no guarda l'estat de la connexió i necessita mecanismes addicionals a la capa d'aplicació (Barth, 2011) per aconseguir el mateix efecte. La utilització de *cookies*[2] és la solució més emprada però també s'han provat efectiu el reconeixement d'empremtes de l'explorador (Eckersley, 2009).

En particular, els motors de cerca implementen aquest tipus de tècniques. A partir d'un anàlisi de les polítiques de registre de *logs* de Google (Toubiana i Nissenbaum, 2011) i la última informació publicada per la companyia al respecte (Google, 2012), sabem que els identificadors registrats per Google en els *logs* del servidor són:

1. Número identificador de la sessió
2. L'adreça IP
3. El timestamp
4. Termes de la *query*
5. *User-agent*[3].

El mateix informe revela que el número identificador de la sessió s'emmagatzema en les *cookies* i és el principal mecanisme usat per Google per a rastrejar els usuaris i detectar sessions.

[2] El protocol introdueix un camp en l'especificació anomenat *Cookie* on s'inclou informació de l'estat de la connexió. Les *cookies* estan dividides en camps. Normalment tenen un camp pel nom i, a vegades, un identificador de la sessió al lloc web.

[3] Aquest terme es pot referir tant al terminal utilitzat per a navegar per la Web com el camp del protocol HTTP que el descriu. Normalment utilitzarem el segon significat. Aquest camp sol incloure el nom de l'explorador, la família, la versió i el sistema operatiu sobre el que està instal·lat, entre d'altres.

Com el Director de Gestió de la Producció de Google, Jack Menzel, clarifica en l'entrevista Menzel (2011), Google inverteix molts esforços en personalitzar usuaris que tenen un compte d'usuari, mentre que els recursos dedicats a usuaris que no disposen d'un compte són molt menors. Les dades d'aquests usuaris només es desen durant 180 dies i trobar patrons es torna una tasca complicada. L'agent es centrarà en protegir els usuaris que no han entrat al seu compte perquè, d'altra manera, ja accepten ser identificats.

Malgrat això, la personalització en anuncis és fa igual per a tots els usuaris. La *cookie* de *doubleclick.net*, anomenada *id*, està associada a un perfil de la *Display Network*, la comunitat de llocs webs on poden aparèixer anuncis d'Ad-words[4]. Per tal d'oferir publicitat personalitzada, aquesta *cookie* s'utilitza per a rastrejar visites *cross-domain*[5] i crear perfils d'usuari.

Per tal de generar una identitat virtual per a cada context, necessitem generar *cookies* i amagar o emmascarar els elements identificadors que hem citat. A continuació descriurem quina és l'estratègia que seguirem en cadascun d'ells.

En primer lloc, generarem dos tipus de *cookies*, unes pel domini de Google i altres pel domini de Doubleclick.net. Per una banda, per generar una nova *cookie* a Google, l'agent estableix una nova connexió amb el camp *Cookie* buit a la capçalera HTTP. Així el servidor interpreta que un nou usuari s'està connectant per primer cop i s'emeten dues *cookies*: *PREFS* i *NID* (Milly, 2004), les quals contenen el número identificador únic del registre de l'usuari (Vegeu l'Annex 7). Per altra banda, l'adquisició de la *cookie id* és més difícil perquè els continguts de la pàgina `www.google.com/ads/preferences` són carregats dinàmicament i l'enllaç per activar la publicitat personalitzada conté un *hash* que canvia en cada nova connexió. El que farem és llegir l'estructura DOM de la pàgina quan estigui completament carregada i, tot seguit, llegir la URL que conté el *hash*. Finalment, quan connectem amb aquest nou recurs, obtindrem la *cookie* desitjada. Els dos tipus de *cookies* són emmagatzemades al disc, en el context que correspongui.

En segon lloc, observem que la IP no és tant útil quan es tracta de rastrejar a l'usuari ja que pot haver-hi centenars o milers d'usuaris navegant darrere un mateix *router*. No obstant, pot donar informació de la localització geogràfica de l'usuari. Per protegir això es pot considerar la utilització de protocols a la capa de xarxa com ToR (The ToR Project, 2002), del qual ja hem parlat a la introducció.

En tercer lloc, l'*user-agent* dóna informació única sobre l'explorador tal i com s'ha demostrat a Eckersley (2009). L'estratègia que segueix l'agent és canviar-lo per un de

[4]Servei de publicitat ofert per Google.
[5]Seqüència de visites a través de diferents dominis d'Internet.

més genèric, per exemple[6]:

```
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11
  (KHTML, like Gecko) Chrome/17.0.963.56 Safari/535.11
```

Val a dir que aquesta cadena de text queda desactualitzada molt ràpidament, per exemple, a cada nova versió de l'explorador.

En últim lloc, el timestamp. Creiem que amb el número de cerques que es fan per minut (vegeu la Introducció), no és necessari emmascarar el timestamp.

4.4 Filtres de *queries*

El DisPA implementa dos filtres. Un filtre per a la freqüència de la *query* del qual ja s'ha parlat en el capítol 3 i un filtre per a les entitats amb nom.

En primer lloc descriurem el filtre de freqüència. Per estimar la freqüència d'una *query* farem servir el mateix cercador que utilitzàvem en el mòdul classificador. Així, suposarem que la freqüència de la *query* és directament proporcional al número de *hits* obtinguts en la cerca. Per aproximar la freqüència mínima f_0 agafarem una *query* que sabem que té baixa freqüència, per exemple, el DNI de l'usuari. Per tant, totes les *queries* per sota d'aquesta freqüència seran considerades consultes que comporten una reducció dràstica de l'anonimitat. En aquest cas hem decidit informar a l'usuari i consultar-li si vol procedir a enviar-la. Si accedeix, la enviarem amb una identitat virtual creada especialment per aquella cerca. Així queden resolts els dos problemes que ens plantejàvem al final del capítol 3.

En segon lloc, crearem un filtre per a les entitats amb nom. La justificació d'aquest filtre és que hi ha termes dins de les *queries* que són independents de la classificació semàntica i poden ser utilitzats per a relacionar *queries* de diversos registres del servidor. Si els termes són poc freqüents (només els té l'usuari), un presumpte atacant els pot fer servir per enllaçar els *logs* i reconstruir la identitat de l'usuari. Observem que això pot passar pel fet que permetem a l'usuari enviar *queries* poc freqüents.

En aquest projecte suposarem que els termes que no són independents de categories semàntiques són els noms propis. La raó és que els noms propis poden ser utilitzats per anomenar qualsevol tipus d'entitat tot i que hi ha convencions, per exemple, pels noms de persones. Tot i així, en general no donen informació sobre l'entitat

[6] Aquest *user-agent* va resultar ser el més genèric d'una enquesta que varem fer i que és accessible a http://www.iiia.csic.es/~mjuarez/results.html. Els resultats van demostrar que aquest *user-agent* havia estat utilitzat 11 cops sobre 87. No és una mostra representativa però era només per a fer un tanteig.

en sí i és difícil classificar-los en una categoria semàntica (excepte noms de personatges famosos, però llavors no són queries poc freqüents). En cas de classificar un nom propi, els termes que l'acompanyen determinaran la categoria semàntica de la consulta. Per exemple, la *query* q = "basket UAB Albert Monzó" va a c = "Esports" i la *query* q = "Albert Monzó sindicat UAB" pot anar a c = "Societat". Si aquestes dues queries es troben en *logs* del servidor diferents, un atacant podria decidir que els dos *logs* són d'un mateix estudiant de l'Autònoma. Com menys freqüents siguin els termes que apareixen a la *query*, més seguretat té l'atacant d'encertar.

En aquest treball ens preocuparem especialment dels noms propis de llocs i persones. L'agent utilitzarà un mòdul pel Reconeixement d'Entitats amb Nom (*Named Entitiy Recognition*) (The Stanford NLP Group, 2011) que, per a cada *query*, detectarà les entitats mencionades en ella. Llavors, tal i com hem comentat en la secció 4.2, utilitzarem contextos diferents per a cada conjunt d'entitats amb nom que ens aparegui. De fet, si fixem un d'aquests conjunts, les queries també poden anar a categories diferents. Així que, de fet, és com si féssim una nova dissociació DisPA per a les queries que continguin el mateix conjunt d'entitats amb nom. L'objectiu és que els termes problemàtics quedin aïllats i els *logs* dissociats del servidor mantinguin la distància mitjana baixa.

4.5 Sistemes de *caching*

Tal i com enunciàvem al principi del capítol 3, una de les hipòtesis del nostre model és que les *queries* només es troben un sol cop al registre de l'usuari. Això és perquè creiem que les repeticions de *queries* també donen molta informació sobre l'usuari. A continuació expliquem la raó.

Segons la intenció de l'usuari, normalment es distingeixen els següents tipus de *queries*.

Queries Informatives: l'usuari cerca sobre un tema extens. Solen haver-hi diferents webs implicades. Per exemple: *cars*, *football*, *england*, etc.

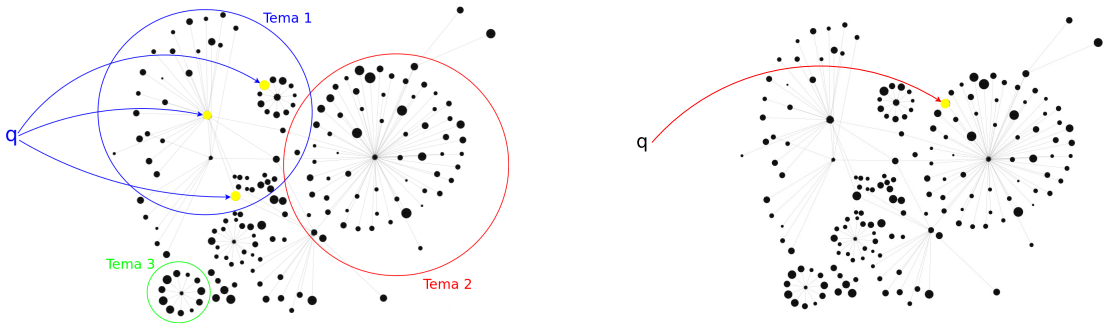
Queries Navigacionals: l'usuari cerca un únic lloc web, la *query* serveix per a dirigir-lo. Exemples d'aques tipus són: *google*, *facebook*, *new york times*, *wikipedia*, etc.

Queries Transaccionals: l'usuari té la intenció de realitzar una operació a Internet. La *query* el dirigeix a diferents *websites* que li permeten realitzar aquesta transacció. Per exemple: *download film*, *buy song*, etc.

Queries de Connectivitat: permeten comprovar la connectivitat entre diferents nodes del graf indexat d'Internet. La majoria de cercadors no permeten fer aquest tipus de

queries.

Creiem que les *queries* informatives i les transaccionals són les més adequades per ser objectiu de l'agent DisPA. La justificació és senzilla, són les que més contingut semàntic tenen (les transaccionals tenen verbs i les informatives, noms), per tant, es poden classificar fàcilment dins d'una taxonomia. En canvi, les *queries* navigacionals són difícils de classificar perquè contenen noms propis de llocs web. A més a més, hem observat que hi ha una tendència dels usuaris a cercar manualment *websites* que utilitzen freqüentment. Per exemple, per tal d'accedir a *http://www.facebook.com*, cerquen a Google la consulta $q =$ "facebook". Això també passa amb la web de la feina, la web de la Universitat, etc. Un atacant podria deduir una relació entre les cerques que més es repeteixen i l'usuari i, això, suposa un altre risc per a la seva privadesa. Per solucionar aquest problema farem servir un sistema de *caching* amb una política LRU[7]. Un cop s'envïi una *query*, desarem els resultats que retorna el cercador a disc i els recuperarem els següents cops que es faci la mateixa *query*, sense haver de connectar amb Google de nou.



(a) Informatives. (b) Navigacionals.

Figura 4.3: Tipus de *queries*

Per altra banda, tal i com havíem anticipat al capítol 2, els resultats també són emmagatzemats localment i els enllaços, tractats (es mostra la URL de la pàgina de destí, traient les redireccions que afegeix Google per registrar informació). L'objectiu és que

[7] Algorisme de reemplaçament que descarta els elements que menys s'han utilitzat primer.

Google no pugui conèixer quins enllaços ha clicat l'usuari a través dels scripts que executa a les pàgines de resultats. Per aquestes raons, l'agent també implementa un sistema de *caching* pels contextos generats. A més, la informació que es desa en els nodes de la taxonomia (pàgines web i *queries* classificades) també es fa persistent i es desa a disc cada cop que els processos de l'*add-on* o l'explorador moren.

4.6 Algorisme DisPA

Finalment, l'algorisme que utilitzarem és:

Algorisme 1 DisPA

Entrada: q *query* a classificar.
Sortida: $\{l_1, \ldots, l_r\}$ enllaços dels resultats.

```
1:  Inicialització
2:  si f(q) < f_min aleshores
3:     Demanar a l'usuari si vol continuar.
4:  si no
5:     si ∃ q aleshores
6:        {l_1, ..., l_r} ← load_results(q)
7:     si no
8:        c ← PQC(q)
9:        {e_1, ..., e_m} ← NER(q)
10:       idv ← gen_virtualID(c, {e_1, ..., e_m})
11:       si ∄ context[idv] aleshores
12:          gen_context(idv)
13:       fi si
14:       {l_1, ..., l_r} ← download_results(context[idv])
15:       store(q,{l_1, ..., l_r})
16:    fi si
17: fi si
```

La funció *NER*() es refereix al filtre d'entitats amb nom i la funció *PQC*() a la classificació personalitzada de la query. La funció *gen_virtualID*() genera una nova identitat virtual (creant noves *cookies*, assignant un *user-agent* adequat, etc.). La funció *gen_context*() genera un nou context associat a una identitat virtual tal i com s'ha definit en les seccions anteriors. *download_results*() connecta a Google utilitzant un context adequat i descarrega els resultats. Finalment, *store*() i *load*() serveixen per emmagatzemar i carregar resultats de la *cache* associats a una cerca.

La complexitat computacional de l'algorisme es redueix a la complexitat de la cerca que, tenint en compte que la llibreria Lucene utilitza el *Vector Space Model*[8] amb índexs

[8]Un model de representació dels documents en forma vectorial, on les posicions dels vectors són les freqüències dels termes.

inversos, aquesta complexitat és la complexitat de l'algorisme de cerca en l'índex invers. Les altres operacions (e.g. càlcul dels pesos per a cada document) són menyspreables. Per tant, en el pitjor cas, aquesta complexitat és

$$\mathcal{O}(|Q| \times |T| \times |D|),$$

on Q són els termes de la *query*[9], T és el conjunt de termes (files de l'índex) i D és el conjunt de documents (columnes de l'índex). Observem que aquesta complexitat és molt més baixa que la complexitat dels protocols PIR tradicionals aplicats a la Web ja que, en comparació, el número de termes i documents indexats és molt petit.

Paral·lelament a aquest algorisme, en segon pla, també classifica dins al taxonomia totes les pàgines web que l'usuari visita. Aquests llocs webs s'utilitzen, tal i com hem vist, per aprendre les preferències de l'usuari segons el model probabilístic que ja hem presentat.

[9]Una *query* té de mitjana 3 termes.

Capítol 5

Desenvolupament

Les extensions dels navegadors són dependents de la plataforma i com que un dels objectius del projecte és que la solució resultant sigui el màxim portable possible, hem desacoblat la lògica de l'agent i la petita part d'interfície que té, en un *back-end* i un *front-end*, respectivament. En el back-end s'implementarà el model tal i com hem vist en la secció anterior. Per aquesta part s'utilitzarà el llenguatge *Java* per tal que sigui multi-plataforma. En contrast, el front-end és l'*add-on* de el navegador i només podem utilitzar un llenguatge que executi el navegador. En particular, el navegador que utilitzarem serà el Firefox, per ser un navegador molt utilitzat i per les comoditats que proporcionen als seus desenvolupadors. Per aquesta raó, el front-end es mantindrà el més reduït possible i la majoria de la lògica es delegarà al back-end. Així també reduïm el cost de fer un *add-on* per un navegador diferent. Ambdós es comuniquen entre sí en una arquitectura de client-servidor, seguint un protocol que hem definit. Val a dir que hem utilitzat les llibreries *JUnit* per a dissenyar testos unitaris. Com a complement hem empleat eines de *coverage* per veure les parts de codi que cobrien els testos.

5.1 L'*Add-on*

Per desenvolupar l'*add-on* hem utilitzat l'Add-on SDK de Firefox (també conegut com a Jetpack). L'*add-on* és molt senzill, bàsicament és una interfície web per tal que l'usuari enviï les queries a través de l'agent. La pàgina web executa uns scripts que escolten les accions de l'usuari i, en el moment que l'usuari envia una *query* (fent clic a "Enviar" o prement la tecla "Intro"), la *query* és enviada al servidor (*framework*), on és processada amb l'algorisme DisPA. Aleshores, els resultats obtinguts de Google s'envien en resposta cap a l'*add-on*. La comunicació entre servidor i client es fa via protocol TCP a través de la interfície de *loopback*.

L'*add-on* també té un mòdul que escolta les webs que visita l'usuari. Tant les que accedeix a través de clics, com les que accedeix introduint la seva adreça URL en el navegador. Aquestes webs també són enviades al servidor on també són processades i classificades.

Com que hi ha diversos tipus de missatges entre client i servidor hem dissenyat un petit protocol molt simple de comunicació. Queda resumit en la taula següent.

Tipus Missatge	Codi d'operació	Missatge
Query	00	Termes de la *query* separats per espais.
Visit	01	URL de la visita.
Error	02	Missatge de text de l'error.
Off	03	Missatge per apagar el servidor.
Results	04	Resultats de la cerca.

Els codis d'operació es troben separats pels missatges amb una barra vertical. Els missatges de resultats sempre van de servidor a client, l'error pot ser tant del client com del servidor i, la resta, sempre van de client a servidor.

5.2 Framework

El *framework* està dissenyat per a ser el més flexible possible. Permet afegir nous models de classificació i diverses maneres de fer la cerca sobre el corpus de l'ODP.

5.2.1 Disseny

A continuació farem una breu explicació de les classes principals del *framework*.

- El paquet *taxonomy* conté tres classes que segueixen el patró de disseny *Composite* i implementen l'estructura de dades per a la taxonomia. Els components de la taxonomia es construeixen a partir de la classe abstracta *TaxonomyElement*, que té dues classes filles. Les fulles de la taxonomia són els *WebSites* i, les classes composades, són les categories construïdes a partir de la classe *Category*. Aleshores, tenim la classe *Taxonomia* que conté una instància d'una categoria anomenada *root* i representa l'arrel de la taxonomia. A més a més, conté mètodes per carregar les dades de l'índex de la taxonomia de l'ODP generat amb el Lucene. També té mètodes per emmagatzemar-la i carregar-la del disc dur.

- El paquet *search* conté les classes necessàries per fer la cerca de les queries localment. Segueix un patró de disseny *Bridge* per separar l'abstracció de la implementació. La

implementació és la manera de generar la *query* pel Lucene. Hi ha dues maneres de generar la *query* depenent de si es tracta de cercar una web o una consulta a Google. Per les consultes crearem una *query* pel Lucene de la manera estàndard. En canvi, si és una web, la *query* es generarà a través de les paraules clau que s'obtenen dels *meta-tags* del codi HTML. En cas de no trobar paraules clau ni descripció, s'agafaran els continguts i es processaran per extreure els trossos de text més representatius. Per fer això últim utilitzem una llibreria de codi lliure (Kohlschütter, 2006-2012). L'abstracció té un fill per cada tipus de cerca. Per ara només hi ha desenvolupada la cerca que construeix la consulta ampliada per WordNet i una cerca simple que construeix la *query* sense ampliar-la. Com veiem, el mètode *extendQuery()* és un *template method* que s'instancia de forma diferent en cada fill.

- El paquet *classifier* conté les classes per classificar la *query* en una categoria. Per aquesta raó conté una instància de la classe taxonomia i una de la classe cercador. La classe *Classifier* conté el mètode *classify*() que té com entrada una *query* de l'usuari (que rep des del client, l'*add-on*) i retorna una categoria. Com hem dit, la *query* pot ser un lloc web o una consulta normal. Les classes *QueryClassifier* i *WebClassifier* hereten de la classe abstracta *Classifier* i tenen un *template method* per afegir un element diferent a la informació del node de la categoria resultat. En cas del *website* s'incrementa en 1 el nombre de queries que han caigut en aquella categoria i, en el cas de la *query*, s'incrementa el número de *sites*.

- El paquet *contexts* té les classes per gestionar els contextos de les connexions HTTP. Tenim les classes *Context* i *Virtual Identity* que defineixen un Context i una Identitat Virtual respectivament. D'una banda, tenim la classe *ContextManager* que conté una instància de la classe *NEDetector*. Aquesta última classe utilitza la llibreria de Standford de reconeixement d'Entitats amb Nom (The Stanford NLP Group, 2011) per detectar noms propis de persones i llocs a la *query*. A més a més, té una instància de *Classifier* per tal d'obtenir la categoria de la *query*. D'altra banda, en el paquet *virtualIdentity* tenim la classe *VirtualIdentityGenerator* que conté el mètode *generateVirtualIdentity()*, per crear noves identitats virtuals. La classe *Context* conté informació de la connexió i una identitat virtual. Finalment, el *ContextManager* conté mètodes per gestionar els contextos. Per a crear sessions i gestionar les *cookies* utilitzem la llibreria d'Apache HTTPComponents (The Apache Software Foundation, 2005-2012). Per implementar la *cache* utilitzem la la llibreria *JCache* (Karlstrøm, 2002-2009).

- En el paquet *queries* trobem les classes necessàries per fer els *requests* a Google. La classe *ResultFecther* conté el mètode *fetch()* que recupera els resultats que retornaria Google utilitzant un context o un altre.

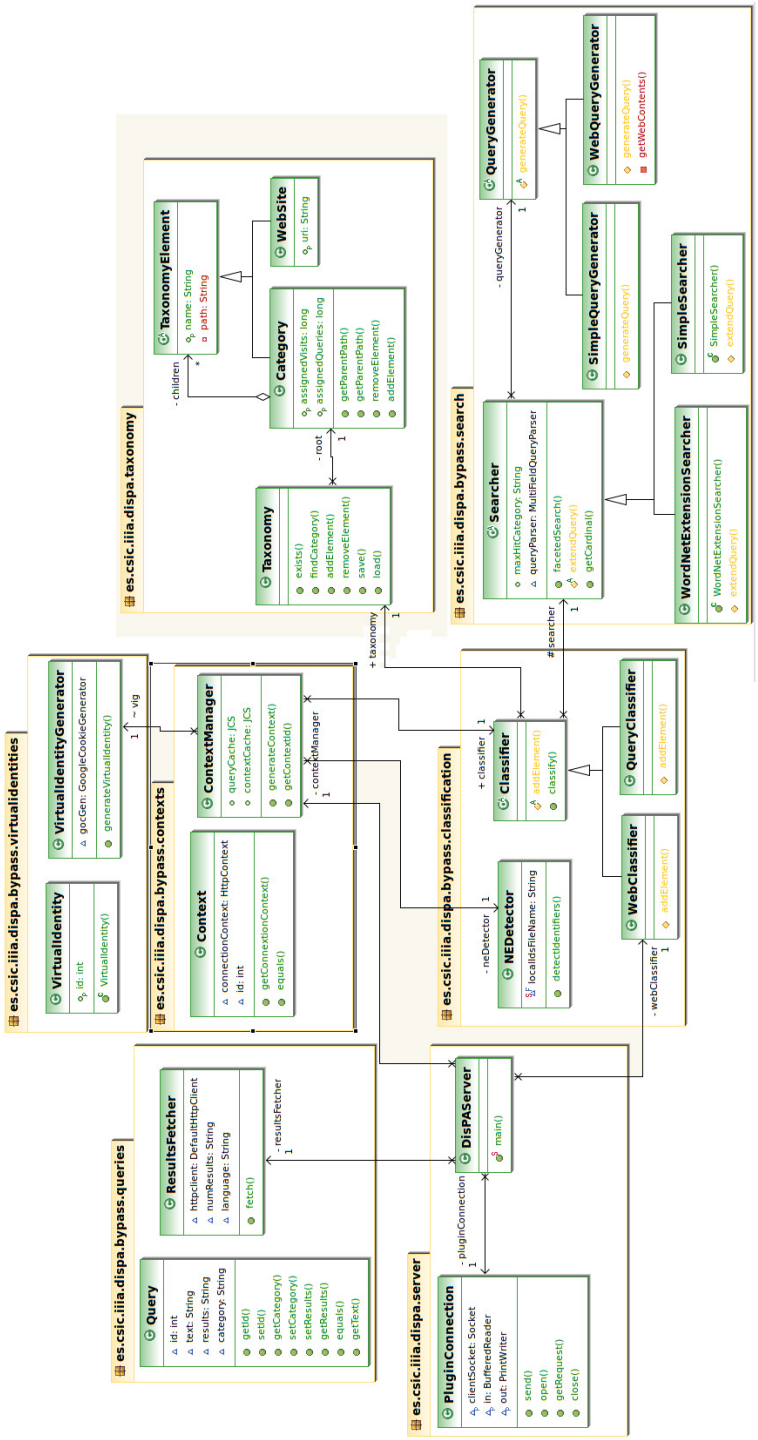


Figura 5.1: Diagrama de classes simplificat del *framework*.

- Per últim, el paquet *server* conté les classes per construir el servidor de l'agent DisPA. Conté la classe *DisPAServer* que implementa l'algorisme de l'agent utilitzant instàncies de les classes *ContextManager*, *WebClassifier* i *QueryFetcher*. També té la classe *PluginConnection* per comunicar-se amb l'*add-on* del Firefox. A partir de l'objecte creat amb aquesta classe rebrà les queries i les webs que s'han de classificar i, també, enviarà els resultats en el sentit contrari.

5.2.2 Avaluació de l'agent: estratègia atacant

L'avaluació del risc de revelació es farà mitjançant un algorisme de *reidentificació*, imitant els algorismes de *record-linkage* utilitzats en la disciplina de *Disclosure Control*. L'algorisme de *reidentificació* pot ser utilitzat per un possible atacant que tingui accés al servidor. Té com a objectiu reconstruir el registre original a partir dels registres parcials resultants de la dissociació.

Imaginem que l'usuari envia termes amb freqüència molt baixa que no queden classificats en una sola categoria i queden escampats per tots els *logs* de l'usuari. El presumpte atacant pot utilitzar aquests termes per tornar a unir tots els registres i recuperar la identitat de l'usuari. Com hem vist en el capítol 4, un exemple d'aquests termes són les entitats amb nom.

La idea de l'algorisme atacant és expressar els *logs* en forma de vectors de termes utilitzant l'esquema *tf-idf*[1]. Aquest esquema reflecteix la importància d'un terme dins d'un registre normalitzat per la freqüència del terme al servidor. Així, si un terme és molt freqüent en el servidor, per molt que ho sigui en un document no tindrà tanta importància com un terme que és menys freqüent en el servidor. Aleshores, l'algorisme agafa aquests vectors com a exemples per a una algorisme de *clustering*. L'algorisme que hem utilitzat és el DBSCAN utilitzant la distància del cosinus com la mesura de similitud entre els exemples. Aquest algorisme de *clustering* està basat en la densitat, accepta soroll, és aglomeratiu i no necessita que se li especifiqui el número de particions final. Només té dos paràmetres: la distància de veïnat i el mínim número d'exemples que té un *clúster*. El funcionament bàsic és el següent. Al principi cada exemple és un *clúster* i, a cada iteració, s'afegeixen tots els exemples que es trobin dins de l'entorn definit per la distància de veïnat. L'algorisme atura quan no es pot afegir cap nou exemples i els *clústers* que no han obtingut el número suficient d'exemples queden com a soroll. Per aplicar aquest *clustering* a l'algorisme de *reidentificació* hem escollit un mínim d'un exemple. La distància de veïnat l'hem deixat com a paràmetre que ha de conèixer l'atacant

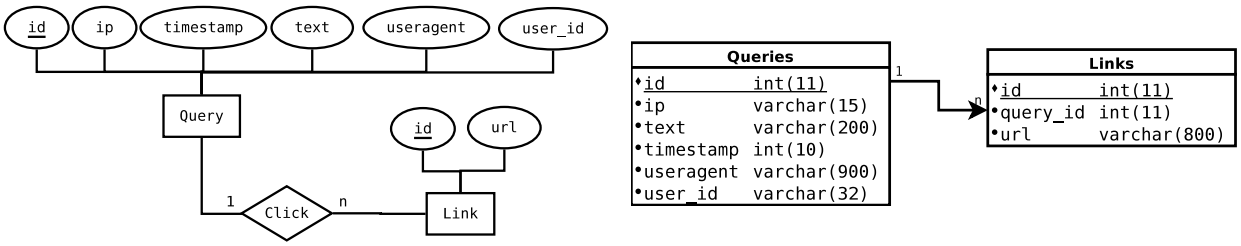
[1] Term Frequency-Inverse Document Frequency.

a priori. A més a més, l'atacant també ha de conèixer com a mínim un *log* parcial del registre que sigui de la víctima. Aquest *log* farà de llavor de manera que, quan l'algorisme de *clustering* pari, els *logs* que hagin caigut en el mateix *clúster* que la llavor es consideren registres de l'usuari atacat. Considerarem el registre original com la unió d'aquests *logs*.

Aquest algorisme ens servirà per avaluar el Risc de Revelació de l'agent. El que farem serà mersurar la qualitat de la partició feta per l'algorisme de *clustering*. Per a fer això, calcularem la $F1$-*score* de la classificació binària definida per la propietat de pertànyer o no al *clúster* final. Aquesta ens donarà una mesura del risc de revelació. No obstant, la $F1$-*score* no la mesurarem pel número d'exemples sinó pel número de queries. Així, els veritables positius són les queries del *log* original que cauen al *clúster* final, els falsos positius són les queries d'altres usuaris que cauen en el *clúster* final, els veritables positius són les queries d'altres usuaris que cauen fora i, els falsos negatius, són les queries del *log* original que cauen fora.

5.2.3 Complements: *queryCollector* i *querySubmitter*

A més, hem programat alguns *add-ons* de Firefox per intentar avaluar la personalització. També estan disponibles a la web del projecte. A continuació farem un petit resum de cadascun d'ells.



(a) Model Conceptual (E-R) (b) Model Relacional.

Figura 5.2: Petit disseny de la base de dades.

- El *queryCollector* recull queries dels voluntaris que acceptin que les seves queries

siguin registrades. A través d'una pàgina web i a través de l'*add-on* (aquest últim fa servir el mateix script php de la web). Les queries són enregistrades en una base de dades MySQL el disseny de la qual es mostra en la Figura 5.2.

- El *querySubmitter* és un *add-on* que, donat un registre de queries, les envia al cercador creant una còpia d'aquest registre al servidor de Google. El principal inconvenient és que Google analitza el trànsit que li arriba de les diferents IPs i si detecta un trànsit anormal fa saltar un *Captcha* per comprovar que no es tracta d'un bot. Per aquesta raó l'*add-on* envia les queries amb intervals de temps aleatoris i que es poden personalitzar. A més a més, si el registre de queries conté el link que l'usuari va clicar, l'*add-on* el cerca a la llista de resultats i, en cas de trobar-lo, el clica.

Capítol 6

Experiments

La falta de *datasets* públics de queries fa difícil l'avaluació del grau de personalització aconseguit per l'agent i, també, la comparació entre utilitzar i no utilitzar l'agent. S'han realitzat proves superficials utilitzant els *add-ons* QuerySubmitter i QueryCollector en un conjunt de 800 queries d'un dels usuaris d'AOL i retorna les pàgines adequades. Un altre experiment amb 2.743 queries d'un altre usuari d'AOL va mostrar que entre cerques personalitzades i no-personalitzades només canviava l'ordre d'un resultat (de la primera a la segona posició.). Malgrat tot, una anàlisi completa de la personalització està fora de l'abast d'aquest projecte. Per això, en la secció d'experiments ens centrarem en avaluar el Risc de Revelació.

6.1 *Datasets*

Hem utilitzat els *logs* d'AOL alliberats per realitzar els experiments. Aquest *dataset* conté aproximadament 20 milions de queries de 650.000 d'usuaris registrades en un període de 3 mesos. Les dades estan desades amb un ID anònim d'usuari i inclou els següents camps:

AnonID: l'identificador d'usuari anònim.

Query: la query enviada per l'usuari.

QueryTime: el timestamp de l'enviament de la query.

ItemRank: si l'usuari va clicar en un resultat, la seva posició en el *ranking* de resultats.

ClickURL: si l'usuari va clicar en un resultat, el domini de la URL del resultat.

El copyright és d'AOL i pot ser utilitzat per a recerca sense finalitats comercials.

6.2 Plantejament

Hem dividit els experiments en dues parts. Per la primera part de l'experiment hem escollit 20 usuaris d'AOL i hem aplicat DisPA sense el filtratge d'entitats amb nom en un d'aquests *logs*. Després hem aplicat l'algorisme atacant amb una llavor aleatòria per avaluar el *Disclosure Risk*.

Per la segona part hem fet dos experiments. Primer hem escollit un conjunt de 20 usuaris i hi hem afegit el *log* de la Thelma Arnold, l'usuari que va ser identificat, i hem aplicat altre cop DisPA sense el filtratge d'entitats amb nom. Hem utilitzat el *log* de la Thelma perquè conté termes com "Arnold" o "Lilburn" que corresponen a entitats amb nom i són independents d'una categoria semàntica. Aleshores, hem avaluat la dissociació que ha fet DisPA amb l'algoritme atacant agafant com a llavors el registre associat a la categoria "Arts", un dels registres dissociats més grans. La justificació és que un atacant que té accés al servidor trobarà més informació coneguda sobre l'usuari en el registre més gran. Per últim, pel segon experiment, hem aplicat DisPA sobre el mateix conjunt de 20 *logs* amb el registre de la Thelma però aquest cop utilitzant el filtre d'identitats per nom.

6.3 Resultats

Pel primer experiment hem mesurat el *recall*[1], la *precision*[2] i la *F1 score*[3] (el lector pot trobar més informació a Manning *et al.* (2008)) del *clustering*. En la Figura 6.1 mostrem els valors d'aquestes mesures per diversos valors de ε. Aquest nombre és un paràmetre que utilitza l'algorisme de *clustering* DBSCAN que defineix el veïnat d'un *clúster*. Observem que per petits valors de ε, el *clustering* té màxima *precision* perquè el *clúster* final conté la llavor. Aleshores, totes les queries de la llavor són queries que pertanyen al *log* objectiu i que cauen al *clúster* final (veritables positius) i, atès que no hi ha cap altre registre al *clúster* final, no hi ha queries d'altres usuaris en el *clúster* final (falsos positius). En contrast, el *recall* és molt baix perquè hi ha un munt de server *logs* de l'usuari que cauen fora del *clúster* final (falsos negatius). Això es tradueix a una baixa F1-*score* i, per tant, a un baix *Disclosure Risk*. A mida que augmentem ε hi ha més registres que cauen al *clúster* final. No obstant, el *log* en particular que hem agafat com a objectiu queda ben dissociat per DisPA i l'algorisme atacant per aquesta llavor i salta directament a la situació en la qual la col·lecció completa de *logs* del servidor cau en el *clúster* final. Això vol dir que

[1] És la proporció d'exemples rellevants que es recuperen.
[2] En *Information Retrieval* és la proporció d'exemples recuperats que són rellevants.
[3] És una mesura per avaluar la qualitat del test que considera la *precision* i el recall: $F_1 = \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$

els *logs* de l'usuari no es poden re-enllaçar utilitzant l'algorisme atacant perquè no tenen prou termes poc freqüents en comú.

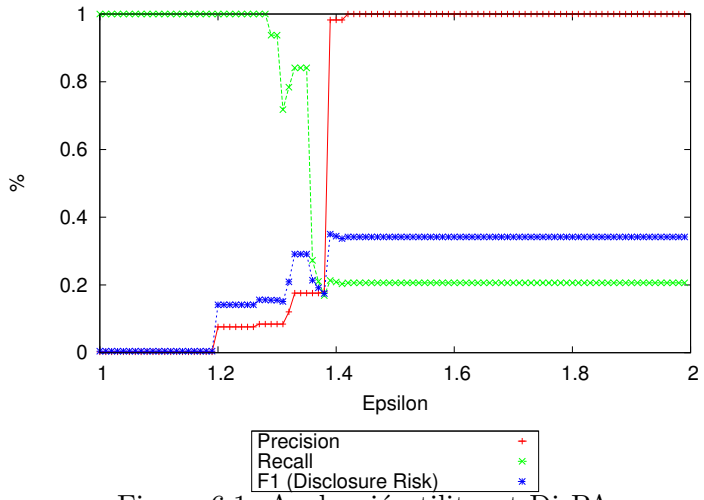


Figura 6.1: Avaluació utilitzant DisPA

En la Figura 6.2 mostrem els resultats del segon experiment. Aquest cop hem utilitzat el *log* de la Thelma Arnold com a objectiu. Com veiem en aquesta figura, per $\varepsilon = 1.39$ tenim un bon *recall* i una bona *precision*, això vol dir que quasi tots els *logs* dissociats de la Thelma cauen al mateix *clúster* final i els *logs* d'altres usuaris cauen a fora. L'algorisme ha aconseguit enllaçar la majoria de *logs* dissociats i, per tant, tenim un alt risc de revelació. Si comparem el registre de la Thelma Arnold amb el registre de l'experiment anterior apreciem que ella fa cerques que contenen termes com "Arnold" o "Lilburn", els quals són poc comuns i cauen en categories diferents depenent dels termes que els acompanyen.

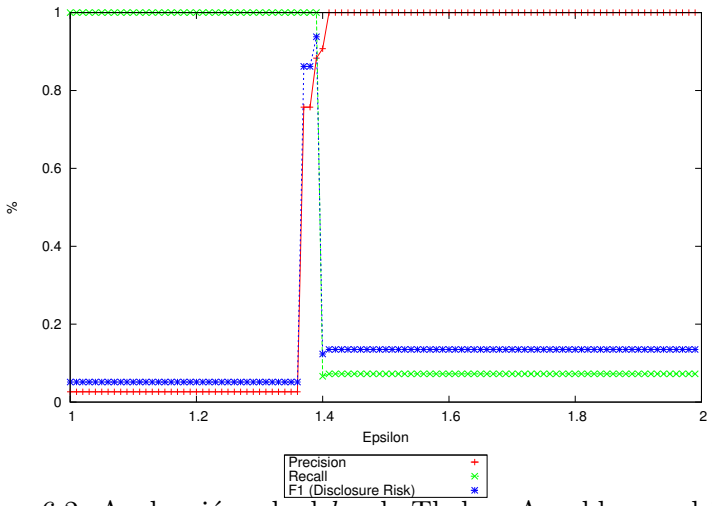


Figura 6.2: Avaluació amb el *log* de Thelma Arnold com objectiu

Per acabar, en l'últim experiment hem pres els mateixos paràmetres per l'algorisme

atacant però aquest cop hem utilitzat el filtre d'entitats amb nom descrit en el capítol 4. En la Figura 6.3 veiem com el risc de revelació augmenta. Per un costat, observem que la *precision* és màxima pels valors més grans de ϵ el que vol dir que el *clúster* conté tots els *logs* dissociats; per altre costat, el *recall* ha disminuït i això vol dir que també conté molts *logs* d'altres usuaris. Per aquesta raó, sabem que l'algorisme no pot obtenir més bons resultats per molt que augmentem la distància del veïnat. Això vol dir que els *logs* no es poden enllaçar perquè els termes poc comuns s'han aconseguit correctament en *logs* diferents.

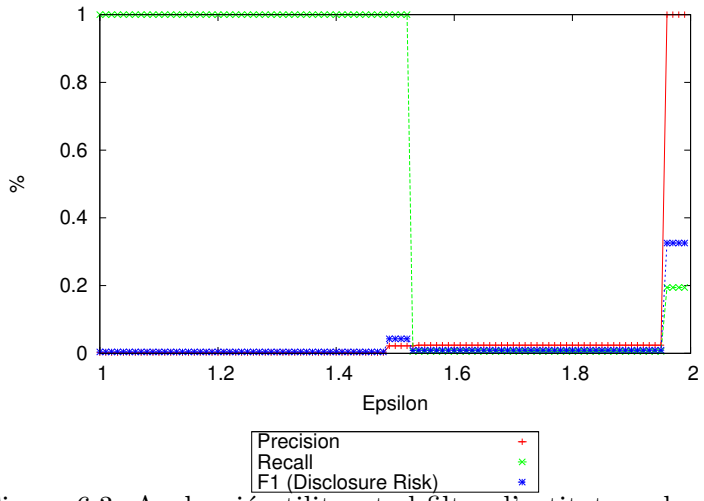


Figura 6.3: Avaluació utilitzant el filtre d'entitats amb nom

Així doncs, l'agent aconsegueix evitar una re-identificació de la Thelma Arnold mitjançant l'algorisme de re-identificació atacant que hem dissenyat en aquest projecte.

Capítol 7

Conclusions i treball futur

La principal contribució d'aquest projecte és la implementació d'un agent que disminueix el risc de revelació de l'usuari en les cerques a la Web i, a més a més, conserva la personalització dels usuaris que no estan autenticats amb un compte d'usuari. Ho aconsegueix sense cooperació per part dels servidors i amb un temps de resposta admissible. També es dóna un model matemàtic del problema i un mètode per solucionar-lo. Finalment s'ha desenvolupat l'agent en un *add-on* pel firefox i el *framework* d'avaluació amb un algorisme atacant de *reidentificació* per a mesurar el risc d'avaluació. A tot això, s'aporta una implementació flexible i multiplataforma del *framework* orientada pel treball futur en l'agent.

Així doncs, tal i com es pot comprovar, els objectius inicials han estat complerts satisfactòriament.

En quant a la planificació inicial, val a dir que ha estat seguida amb rigorositat i, per tant, tota la feina ha estat realitzada dins dels terminis preestablerts.

Cal destacar que amb el treball realitzat s'ha escrit un article científic que serà publicat en una revista científica (Juárez i Torra, 2013).

De cara al treball futur, hi ha diversos fronts oberts. En primer lloc, es pot estudiar fer una personalització local completa reordenant els resultats. Així no depenem de la personalització que ens ofereix el cercador i és més escalable perquè distribuïm el còmput. Per a fer això s'haurien de tenir en compte els punts que hem descrit al capítol 2.

Una altra possible via és, en comptes de fixar un nivell de la taxonomia, establir les categories que descriuen les facetes de l'usuari de manera dinàmica. Això permetria una adaptació dels interessos a llarg termini i s'aconseguiria reduir més el risc de revelació. Amb aquest tema s'esta elaborant un nou article que es preveu enviar per la seva possible presentació en la conferència internacional "*Privacy Security and Trust*" (PST 2013) que

es celebra aquest estiu.

Una altra estratègia és, quan trobem queries poc freqüents, enlloc de no enviar-les o enviar-les i augmentar el risc de revelació, generalitzar-les per augmentar la freqüència només en aquests casos. Un exemple d'això pot ser substituir "Lilburn" per "Atlanta" o "Georgia". Amb aquest procés es produeix pèrdua d'informació i, conseqüentment, utilitat per a personalitzar, però l'usuari pot trobar els resultats igualment sense comprometre la seva privadesa.

Una altra possible línia de recerca oberta es troba en algorismes de re-identificació del costat del servidor. L'algorisme atacant que hem dissenyat en aquest projecte s'ha de millorar molt. La complexitat de l'algorisme és molt alta i el servidor té molts registres, de l'ordre de milers de milions. També es poden provar altres sistemes de *clustering* apart del DBSCAN com els algorismes de *clustering* seqüencial dissenyats per Miyamoto i Arai (2009). Aquests algorismes permeten extreure un *clúster* darrere l'altre. En el nostre cas només voldríem extreure el primer cluster amb els registres de l'usuari i, així, reduir la complexitat. Per un altre costat, la mesura de similitud pot ser reforçada afegint pes extra als termes que sabem que són més distintius de l'usuari.

Val a dir que com a treball immediat, s'ha d'acabar la documentació del treball, que ara es troba en fase de desenvolupament.

Annex 1: Suplantació d'identitat HTTP

La següent captura de pantalla mostra una connexió HTTP al *host google.com*. Demanem el path *google.com/* utilitzant la comanda HEAD, és a dir, li indiquem que volem ignorar els continguts HTML de la web i quedar-nos tant sols amb la capçalera (Kristol i Montulli, 2000).

```
$ telnet google.com 80

Trying 173.194.34.195...
Connected to google.com.
Escape character is '^]'.
HEAD / HTTP/1.0

HTTP/1.0 302 Found
Location: http://www.google.es/
Cache-Control: private
Content-Type: text/html; charset=UTF-8
Set-Cookie:
PREF=ID=d0b77a545d6688fd:FF=0:TM=1331549764:LM=1331549764:S=IA-Pi-A1G gl5mBvf;
expires=Wed, 12-Mar-2014 10:56:04 GMT; path=/; domain=.google.com
Set-Cookie:
NID=57=S9mSzouwS0GW7Ym93v7XFX2WVZEaxGZfVeg8PVeAKwMVGkVQzImnvLE6-cG58S
bDapwlu06RqHxEc_w9USmdl7C2L4YEk6rH5nS8G940G-PbcBJ7Yl8Dbx8w03DNL3Ot; expires=Tue,
11-Sep-2012 10:56:04 GMT; path=/; domain=.google.com; HttpOnly
P3P: CP="This is not a P3P policy! See
http://www.google.com/support/accounts/bin/answer.py?hl=en&answer=151657 for
more info."
Date: Mon, 12 Mar 2012 10:56:04 GMT
Server: gws
```

```
Content-Length: 218
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
```

Observem que el servidor ens ha entregat una resposta amb instàncies de dues *cookies* mitjançant el camp *Set-Cookie*. Les *cookies* que ens envia no tenen l'atribut *session*[1] i expiren d'aquí dos anys. Per tant, si algú altre les vol utilitzar en una nova connexió, pot emplear la següent capçalera.

```
GET / HTTP/1.1
Host: 173.194.34.242
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML,
like Gecko) Chrome/17.0.963.56 Safari/535.11
Http-Referer: http://www.google.com/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Cookie: PREF=ID=d0b77a545d6688fd:FF=0:TM=1331549764:LM=1331549764:S=IA-Pi-A1G
gl5mBvf; expires=Wed, 12-Mar-2014 10:56:04 GMT; path=/; domain=.google.com
Cookie: NID=57=S9mSzouwS0GW7Ym93v7XFX2WVZEaxGZfVeg8PVeAKwMVGkVQzImnvLE6-cG58S
bDapwlu06RqHxEc_w9USmdl7C2L4YEk6rH5nS8G940G-PbcBJ7Yl8Dbx8w03DNL3Ot; expires=Tue,
11-Sep-2012 10:56:04 GMT; path=/; domain=.google.com; HttpOnly
KVzFG_BByln7SYfh05a-8m4RtTSiaOONlORbmfqEAm8pn5KvjJFU_Vb; expires=Wed, 12 Sep
2012 09:20:26 GMT; path=/; domain=.google.es; HttpOnly
```

Cada camp *Cookie* d'aquest request HTTP conté una *cookie* rebuda en la connexió anterior. El servidor interpretarà que ambdues connexions pertanyen a la mateixa sessió i les queries es registraran en el mateix *log*, o sigui, en el registre associat a l'usuari amb *ID=d0b77a545d6688fd*.

És important tenir en compte el camp *Http_Referer*, ja que ha de ser coherent amb les connexions anteriors. A continuació posem un exemple.

Normalment, per arribar a la pàgina de resultats de Google, abans hem introduït la cerca a la pàgina principal del cercador i, en enviar el formulari, ens ha redirigit a la pàgina dels resultats. En aquest cas l'*Http_Referer* un cop a la pàgina de resultats tindria el valor *www.google.com*. No obstant, també es pot introduir l'adreça URL dels resultats directament i obtenir els resultats d'una cerca fent un request de *http://www.google.com/search?q=*<**terme 1**>*+...+*<**terme N**>. Aleshores, Google pot detectar aquesta connexió incoherent perquè no ve dirigit de la pàgina principal i pot suposar que no es tracta d'una persona, sinó d'un *bot*.

[1]Indica que la *cookie* perd validesa en finalitzar la sessió del navegador.

Actualment Google permet aquestes connexions però és raonable sospitar d'aquest tipus de comportament de l'usuari. Altres camps que poden ser utilitzats per verificar connexions coherents són l'*User-Agent* i l'*Accept*.

Annex 2: Informació sobre les *cookies* de Google

Per cada nova connexió a Google, la *cookie* PREF és entregada a l'usuari per part del servidor. Aquesta *cookie* emmagatzema informació sobre les preferències de la cerca. Hem realitzat un petit estudi sobre els seus camps i l'hem resumit en la llista següent.

Taula 7.1: Camps de la *Cookie* PREF .

Nom	**Obligat**	**Format**	**Exemple**	**Descripció**
ID	Sí	16 Bytes	a0b3ef774198c127	Identificador de *cookie*: una part de l'identificador global (*Globally Unique Identifier*, GUID).
U	No	16 Bytes	d279279d446b343b	Apareix a partir de la segona connexió amb la mateixa *cookie* i està documentat com la segona part del GUID (Toubiana i Nissenbaum, 2011).
FF	Sí	Enter	4	Filtre *SafeSearch*. Posibles valors coneguts: • 0: Moderat • 1: Estricte • 4: Deshabilitat
LD	No	2/3 caràcters	en	Codi de l'idioma de cerca per defecte.
				Continua a la pàg. següent

Nom	Obligat	Format	Exemple	Descripció
LR	No	Anell OR	lang_es — lang_en	Si es selecciona més d'un idioma, el camp LD anterior es reemplaça per aquest camp amb un OR dels llenguatges seleccionats.
NR	No	Enter	50	Número de resultats que es mostraran a cada pàgina (10 per defecte si aquest camp és omès). Valors possibles: 10, 20, 30, 50, 100.
NW	No	1 bit	1	Indica si la cerca ha de mostrar-se en una nova finestra.
TM	Sí	UNIX epoch	1331554197	*Timestamp* de la creació de la *cookie*.
LM	Sí	UNIX epoch	1354546434	*Timestamp* de l'últim canvi de preferències.
GM	No	1 bit	1	No s'ha documentat mai per Google i no apareix sempre. En cas d'aparèixer, sempre ho fa sempre amb valor 1.
SG	No	Enter	2	Configuració del *Google Instant*: • 1: Quan els recursos de l'ordinador ho permetin. • 2: Mai. • 3: Sempre.

Continua a la pàg. següent

Nom	Obligat	Format	Exemple	Descripció
S	Sí	Hash	SgLoY4Q6DUqQk2So	Mai documentat per Google. Canvia quan altres camps ho fan. Possiblement és un *checksum* per validar els altres camps (Conti, 2009).

Hi ha altres camps que s'han pogut trobar en aquesta *cookie* dels quals es desconeix el significat. Google mai els ha documentat públicament (Toubiana i Nissenbaum, 2011). Tot i així, aquests valors solen ser *flags* d'estat i, conseqüentment, no donen gran quantitat d'informació (1 bit).

La majoria de paràmetres citats en la taula anterior poden ser modificats per l'usuari via web a través de la pàgina: `http://www.google.com/preferences`

Bibliografia

Acquisti, A.; Gross, R. (2006): «Imagined communities: Awareness, information sharing, and privacy on the facebook». Dins *Privacy Enhancing Technologies*, Lecture Notes in Computer Science, ps. 36–58. Springer Berlin / Heidelberg. ISBN 978-3-540-68790-0.

Adar, E. (2007): «User 4xxxxx9: Anonymizing query logs». *Proc of Query Log Analysis Workshop, International Conference on World Wide Web*.

Arrington, M. (2006): «AOL proudly releases massive amounts of private data». Tech Crunch. http://techcrunch.com/2006/08/06/aol-proudly-releases-massive-amounts-of-user-search-data/.

Barbaro, M.; Zeller, T. (2006): «A face is exposed for AOL searcher no. 4417749». New York Times. http://select.nytimes.com/gst/abstract.html?res=F10612FC345B0C7A8CDDA10894DE404482.

Barth, A. (2011): «HTTP State Management Mechanism». (RFC6265). http://www.ietf.org/rfc/rfc6265.txt.

Billsus, D.; Pazzani, M. J. (1999): «A hybrid user model for news story classification». ps. 99–108. http://dl.acm.org/citation.cfm?id=317328.317338.

Buys, J. (2010): «DuckDuckGo: A New Search Engine Built from Open Source». OSTATIC. http://ostatic.com/blog/duckduckgo-a-new-search-engine-built-from-open-source.

Cao, B.; Sun, J.-T.; Xiang, E. W.; Hu, D. H.; Yang, Q.; Chen, Z. (2009): «Pqc: personalized query classification». Dins *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, ps. 1217–1226. ACM, New York, NY, USA. ISBN 978-1-60558-512-3.

Chaum, D. L. (1981): «Untraceable electronic mail, return addresses, and digital pseudonyms». *Commun. ACM*, volum 24(2): ps. 84–90. ISSN 0001-0782.

Chen, C.-C.; Chen, M. C.; Sun, Y. (2002): «PVA: A self-adaptive personal view agent». *Journal of Intelligent Information Systems*, ps. 173–194. http://www.springerlink.com/index/x224178u6q54389r.pdf.

Chor, B.; Goldreich, O.; Kushilevitz, E.; Sudan, M. (1995): «Private information retrieval». Dins *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, ps. 41–50. IEEE.

comScore (2009): «comScore reports global search market growth of 46 percent in 2009». http://www.comscore.com/Press_Events/Press_Releases/2010/1/Global_Search_Market_Grows_46_Percent_in_2009.

Conti, G. (2009): *Googling security: how much does Google know about you?* Addison-Wesley, the University of California. ISBN 9780321518668.

Cooper, A. (2008): «A survey of query log privacy-enhancing techniques from a policy perspective». *ACM Trans. Web*, volum 2(4): ps. 19:1–19:27. ISSN 1559-1131.

Domingo-Ferrer, J.; Solanas, A.; Castellà-Roca, J. (2008): «h(k)-private information retrieval from privacy-uncooperative queryable databases».

Domingo-Ferrer, J.; Bras-Amorós, M.; Wu, Q.; Manjón, J. (2009): «User-private information retrieval based on a peer-to-peer community». *Data and Knowledge Engineering*.

Eckersley, P. (2009): «How Unique Is Your Web Browser?» Report tècnic, Electronic Frontier Foundation. https://panopticlick.eff.org/browser-uniqueness.pdf.

EFF (2009): «AOL's massive data leak». Electronic Frontier Foundation. http://w2.eff.org/Privacy/AOL/.

Erola, A.; Castellà-Roca, J.; Navarro-Arribas, G.; Torra, V. (2011): «Semantic microaggregation for the anonymization of query logs using the open directory project». *SORT - Statistics and Operations Research Transactions*, ps. 41–58.

Gentry, C.; Ramzan, Z. (2005): «Single-database private information retrieval with constant communication rate». *Automata, Languages and Programming*, ps. 803–815. http://www.springerlink.com/index/80959djt41b816rc.pdf.

Google (2012): «Key Terms - Policies and Principles». http://www.google.com/intl/en/policies/privacy/key-terms/#toc-terms-server-logs.

He, Y.; Naughton, J. F. (2009): «Anonymization of set-valued data via top-down, local generalization». *Proceedings of the VLDB Endowment*, volum 2(1): ps. 934–945.

Hong, Y.; He, X.; Vaidya, J.; Adam, N.; Atluri, V. (2009): «Effective anonymization of query logs». Dins *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, ps. 1465–1468. ACM, New York, NY, USA. ISBN 978-1-60558-512-3.

Howe, D. C.; Nissenbaum, H. (2011): «Trackmenot: resisting surveillance in web search». http://cs.nyu.edu/trackmenot/.

Juárez, M.; Torra, V. (2013): «Towards a privacy agent for information retrieval». *International Journal of Intelligent Systems*. In press.

Karlstrøm, F. (2002-2009): «JCache». http://sourceforge.net/projects/jcache/files/jcache/.

Kohlschütter, C. (2006-2012): «Boilerplate removal and fulltext extraction from HTML». http://code.google.com/p/boilerpipe/.

Kristol, D.; Montulli, L. (2000): «IETF RFC2965 - HTTP state management mechanism». (RFC6265). https://wiki.tools.ietf.org/html/rfc2965.

Kushilevitz, E.; Ostrovsky, R. (1997): «Replication is not needed: Single database, computationally-private information retrieval». Dins *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, ps. 364–373. IEEE.

Lipmaa, H. (2010): «First cpir protocol with data-dependent computation». Dins *Information, Security and Cryptology – ICISC 2009*, *Lecture Notes in Computer Science*, volum 5984, ps. 193–210. Springer Berlin Heidelberg. ISBN 978-3-642-14422-6.

Liu, F.; Yu, C.; Meng, W. (2002): «Personalized web search by mapping user queries to categories». *Proceedings of the eleventh international conference on Information and knowledge management - CIKM '02*, p. 558.

Manning, C. D.; Raghavan, P.; Schütze, H. (2008): *Introduction to Information Retrieval*. Cambridge University Press. http://nlp.stanford.edu/IR-book/.

Menzel, J. (2011): «How google does personalization with jack menzel». Stone Temple Consulting. http://www.stonetemple.com/how-google-does-personalization-with-jack-menzel/.

Miller, G. (2010): «WordNet-about us. WordNet. Princeton University». http://wordnet.princeton.edu/.

Mills, E. (2006): «AOL sued over web search data release». CNET News. http://news.cnet.com/8301-10784_3-6119218-7.html.

Milly (2004): «Anonymizing google's cookie». http://www.imilly.com/google-cookie.htm.

Miyamoto, S.; Arai, K. (2009): «Different sequential clustering algorithms and sequential regression models». Dins *2009 IEEE International Conference on Fuzzy Systems*, ps. 1107–1112. IEEE. ISBN 978-1-4244-3596-8. URL http://dblp.uni-trier.de/db/conf/fuzzIEEE/fuzzIEEE2009.html#MiyamotoA09.

Montoyo, A.; Palomar, M.; Rigau, G. (2001): «Interface for wordnet enrichment with classification systems». Dins *Database and Expert Systems Applications*, *Lecture Notes in Computer Science*, volum 2113, ps. 122–130. Springer Berlin Heidelberg. ISBN 978-3-540-42527-4.

Navarro-Arribas, G.; Torra, V. (2009): «Tree-based microaggregation for the anonymization of search logs». volum 3, ps. 155–158. IEEE, Milan, Italy. ISBN 978-0-7695-3801-3.

Netmarketshare (2012): «Search engine market share». http://marketshare.hitslink.com/search-engine-market-share.aspx?qprid=4.

Norvig, P. (2011): «Search Algorithms with Google Director of Research Peter Norvig». Stone Temple Consulting. http://www.stonetemple.com/search-algorithms-with-google-director-of-research-peter-norvig/.

Pariser, E. (2011): «Beware online filter bubbles». TED Talks. Video available at: http://www.ted.com/talks/eli_pariser_beware_online_filter_bubbles.html.

Rambam, S. (2006): «Privacy is dead - get over it». Dins *Toor2122*. Video.google.com. http://video.google.com/videoplay?docid=-383709537384528624.

Sadetsky, G. (2006): «AOL Data». http://www.gregsadetsky.com/aol-data/.

Samarati, P. (2001): «Protecting respondents identities in microdata release». *Knowledge and Data Engineering, IEEE Transactions on*, volum 13(6): ps. 1010–1027.

Search-logs (2006): «Leaked AOL search database». Search-logs. http://search-logs.com/.

Shen, D.; Pan, R.; Sun, J.; Pan, J.; Wu, K.; Yin, J.; Yang, Q. (2005): «{Q2C@UST}: Our Winning Solution to Query Classification in {KDDCUP} 2005». Dins {*SIGKDD*} *Explorations*, volum 7, ps. 100–110. ACM.

Shen, D.; Pan, R.; Sun, J.-T.; Pan, J. J.; Wu, K.; Yin, J.; Yang, Q. (2006): «Query enrichment for web-query classification». *ACM Transactions on Information Systems*, volum 24(3): ps. 320–352. ISSN 1046-8188.

Speretta, M.; Gauch, S. (2005): «Personalized search based on user search histories». Dins *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on*, ps. 622 – 628.

statOWL (2012): «Search engine market share». http://www.statowl.com/search_engine_market_share.php.

Sweeney, L. (2002): «k-anonymity: A model for protecting privacy». *International Journal of Uncertainty Fuzziness and Knowledge Based Systems*, volum 10(5): ps. 557–570.

The Apache Software Foundation (2005-2012): «HTTP components». http://hc.apache.org/index.html.

The Stanford NLP Group (2011): «Stanford named entity recognizer (NER)». http://nlp.stanford.edu/software/CRF-NER.shtml.

The ToR Project (2002): «Facebook, a fun resource or invasion of privacy.» https://www.torproject.org.

Titanium, Johnny "Doc Evil"(2006): «AOL search log special». Something Awful. http://www.somethingawful.com/d/weekend-web/aol-search-log.php.

Torra, V. (2007): *Fonaments de la Intel·ligència Artificial*. Fundació UOC.

Toubiana, V.; Nissenbaum, H. (2011): «Analysis of google logs retention policies». *Journal of Privacy and Confidentiality*, volum 3(1): p. 3–26.

Ullegaddi, P.; Varma, V. (2010): «A Simple Unsupervised Query Categorizer for Web Search Engines». *International Institute of Information Technology, India*.

Whelan, B. (2005): «Facebook, a fun resource or invasion of privacy.» Athensnews.com. http://athensnews.com/issue/article.php3?story_id=21491.

WorldNetDialy (2007): «Fed up with Google? Try Scroogle.org. Powerful search tool without privacy violations». http://www.wnd.com/2007/06/41839/.

Firmat: Marc Juárez

Bellaterra, 29 de gener de 2013

Resum

En aquest projecte tractem el problema de Recuperació Privada de la Informació (PIR) associat amb l'ús dels cercadors web. La contribució principal és el desenvolupament d'un agent no cooperatiu que asseguri un nivell de privadesa a l'usuari i preservi la personalització de la cerca proporcionada pels proveïdors del cercador.

Resumen

En este proyecto tratamos el problema de la Recuperación Privada de la Información (PIR) asociado al uso de buscadores web. La contribución principal es el desarrollo de un agente no cooperativo que asegure un nivel de privacidad al usuario y preserve la personalización de la búsqueda proporcionada por los proveedores del buscador.

Abstract

In this project we tackle the Private Information Retrieval (PIR) problem associated with the use of web search engines. The main contribution is the development of a non-cooperative agent that assures a level of privacy to the user and preserves search personalization offered by the search-engine providers.