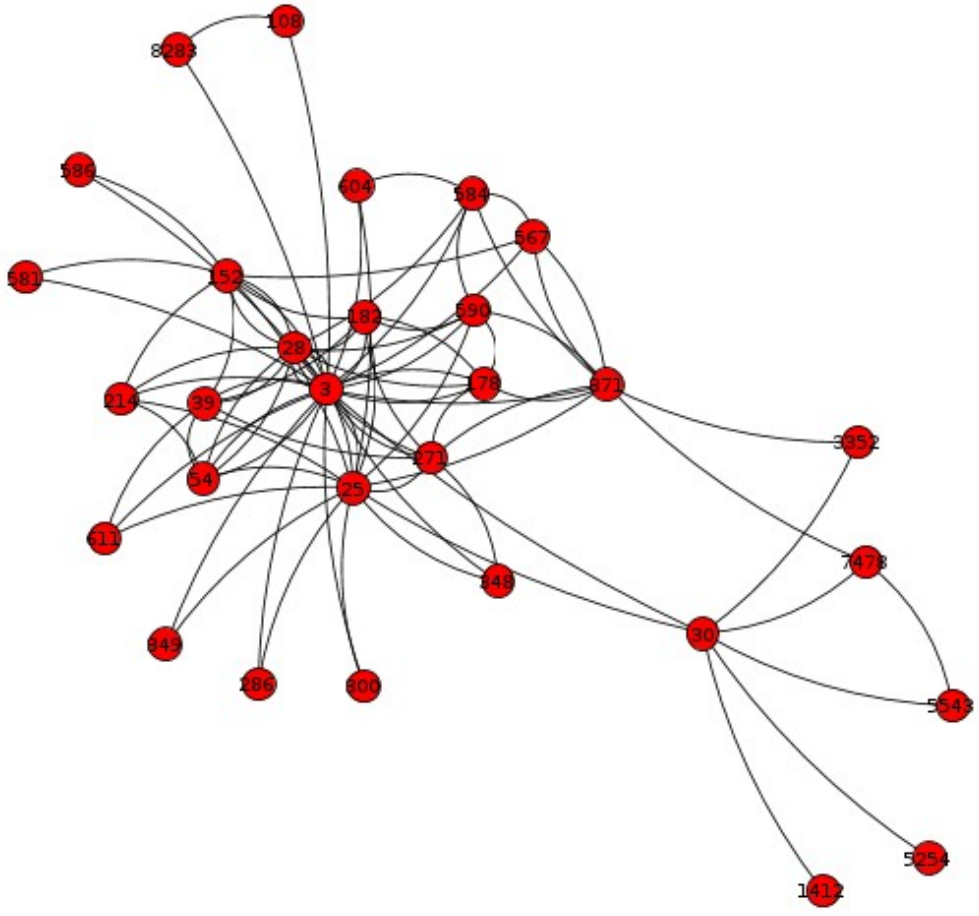
Estudi dels mètodes d'estimació de la tipologia de la xarxa en un protocol UUP basats en xarxes socials.



Josep Canyelles Frau
Enginyeria en Informàtica

Consultor: Jordi Castellà-Roca
03/06/2012

El paper dels cercadors d'Internet dins la nostra “societat de la informació” actual juguen un paper clau ja que cada pic més es valora el fet d'obtenir les dades correctes, en el moment adient i en el format apropiat. Avui en dia, donada la gran quantitat d’informació disponible a Internet, ha fet que el pes dels cercadors web (Web Search Engines) - per exemple Google, Yahoo, Bing,..- en l’ús diari de la xarxa sigui molt alt i hagin esdevingut una eina essencial per a la recerca d’aquesta informació. Aquestes eines tan útils utilitza mecanismes per a obtenir informació dels seus usuaris -el que en diem perfils- per tractar d'encertar i apurar més en la cerca. Aquestes dades són emmagatzema i analitzades pels motors de cerca a tal fi. Estudis han demostrat que mitjançant les consultes fetes pels distints usuaris del cercadors d'informació es poden arribar a perfilar en molt detall aquests usuaris, la qual cosa suposa una greu pèrdua de privacitat. En aquest projecte es presenta un mecanisme per a garantir la privacitat de les cerques que fan els usuaris en front a les perfilats dels motors de cerca, aprofitant del potencial de les xarxes socials.

Índex de continguts

Índex d'il·lustracions

Índex de taules

1 Introducció.

En la “societat de la informació” actual obtenir les dades correctes, en el moment adient i en el format apropiat ha esdevingut un objectiu molt important en tots els sectors. Avui en dia, donada la gran quantitat d’informació disponible a Internet, ha fet que el pes dels cercadors web (Web Search Engines) - per exemple Google, Yahoo, Bing,..- en l’ús diari de la xarxa sigui molt alt i hagin esdevingut una eina essencial per a la recerca d’aquesta informació. Recents estudis mostren que la cerca d’informació mitjançant un WSE i la gestió del correu electrònic continuen essent les dues tasques més populars a Internet. Una enquesta feta per Pew Internet[1] conclou que el 92% dels adult connectats a Internet utilitzen els WSE per a trobar informació en la Web, d’entre ells un 59% que ho fan diàriament.

Gairebé tots els WSE mostren els resultats obtinguts mitjançant una llista d'enllaços distribuïts en vàries pàgines de resultats, disposats de tal manera que els “millors” resultats apareixen en les primeres posicions. Estudis mostren com el 92% dels usuaris seleccionen un resultat dins les tres primeres pàgines retornades pel WSE. Per tal de oferir millors resultats en les cerques que realitzen els usuaris, els WSE solen enregistrar totes les consultes enviades i els resultats que han estat seleccionats. D’aquesta manera els WSE puntuen els resultats de les cerques segons la rellevància que li atorguen els usuaris amb la seva selecció.

Aquesta puntuació de les pàgines no pot per si mateixa resoldre un problema important: l'ambigüitat d’alguns termes. En el procés de desambiguació d’un terme es requereix de certa informació de l’usuari que fa la consulta o del context de la consulta mateixa. Estudis en aquest sentit [15,16], s'han centrat en què els usuaris hagin d'introduir explícitament la seva informació contextual, incloent temes d'interès, favorits,etc. Per altra banda també s'ha prestat molta atenció en la manera de poder aprendre els interessos dels usuaris de forma automàtica mitjançant models de perfils d'usuari [17, 18]. En aquest treball ens centrarem en aquesta darrera forma.

Els interessos d’un usuari es poden agregar formant perfils d’usuari. L’agregació de les cerques fetes per un individu poden revelar molta informació de l’usuari individual o de les institucions per les que treballen. Ja al 1986 [2] es va veure que els sistemes de recuperació de informació pot personalitzar les cerques dels usuaris mitjançant la creació de perfils.

Hi ha diverses tècniques per a perfilar els interessos d’un usuari del WSE: historial de navegació[3], utilització del click-though[4], a traves de l’aplicació de navegació del client[5]. Però el més utilitzat actualment, i el que pareix ser la millor solució, és l’emmagatzemament i anàlisi de les consultes enviades pels usuaris. Amb aquest darrer mecanisme, no es necessita cap col·laboració per part de l’usuari, sinó que la tasca és pròpia del WSE. S’ha estudiat de manera positiva l’aplicació de vàries tècniques de mineria de dades per extreure patrons d’us en aquestes dades emmagatzemades

[6, 7, 8, 9].

Sovint aquests perfils son utilitzats per a millorar les consultes, però també poder ser utilitzats més àmpliament, la qual cosa pot atemptar contra la privacitat dels mateixos. Aquesta situació planteja una greu amenaça pels usuaris i la seva privacitat. Per exemple si un determinat usuari cerca, en diferents moments, informació sobre una malaltia (o sobre algun partit polític, o sobre alguna religió), algú que analitzi els logs del WSE podria inferir que la pateix. Això es va veure molt clar quan, arran de la publicació per part de AOL[14] dels logs de les consultes fetes pels usuaris es varen poder perfilar molts d'aquests, encara que s'hagués reemplaçat la identitat d'aquests (adreça IP) mitjançant una pseudo-identitat. Aquest escàndol va posà de manifest que els WSE no poden protegir correctament la privacitat dels seus usuaris. De totes maneres s'ha d'arribar a un compromís entre el nivell de privacitat cercat i la qualitat del servei del WSE: com més alt sigui aquest nivell, el nivell de qualitat del servei disminuirà (el procés de desambiguació dels termes serà complexe o nul·la), i a l'en revers, si volem una alta qualitat del servei, possiblement haurem de disminuir el nivell de privacitat de l'usuari.

En la literatura sobre aquest tema, es defineix i s'analitza quatre nivells de protecció de la privacitat: pseudo-identitat, identitat de grup, sense identitat, sense informació personal.

En el primer nivell -nivell de pseudo-identitat- la identitat de l'usuari es reemplaçada per una pseudo-identitat que conté menys informació identificable. En aquest nivell els WSE poden perfilar aquestes pseudo-identitats. S'ha comprovat que aquest nivell no és suficient per a protegir la privacitat d'un usuari ja que es permet l'agregació de tota la informació necessària per a descriure'l. Les consultes mostren directament el interessos de l'usuari, si s'agrupen moltes consultes de la mateixa pseudo-identitat, fan possible identificar-lo.[10]

En el segon nivell de privacitat -identitat de grup- un grup d'usuaris comparteixen la mateixa identitat de tal manera que el WSE no pot fer una relació usuari perfil, sinó que solament pot perfilar el grup. Hi ha tres possibles maneres d'implementar aquest nivell de privacitat: utilitzant un proxy per a construir el grup, utilitzant un mecanisme d'ofuscació per enviar consultes aleatòries i per últim enviant consultes generades per altres usuaris. En aquest treball ens centrarem en aquest darrer cas d'identitat de grup, és a dir, a mecanismes que es basen en enviar consultes generades per altres usuaris i concretament als els protocols UUP (Useless User Profile) on l'esquema bàsic consisteix en què cada usuari que vol enviar una consulta al WSE, no l'envia ell mateix sinó que delega aquest enviament a usuaris veïnats de la xarxa. D'aquesta manera, utilitzant aquest mecanisme, el WSE no pot genera un perfil real d'un individu en concret. Per la seva banda, per mitjà de mètodes criptogràfics, els usuaris no saben de qui és la consulta que ha enviat, i per tant, ell tampoc pot fer cap perfil .

En el tercer nivell de privacitat -sense identitat- la identitat dels usuaris no està disponible pel WSE. És a dir s'implementa un procés per enviar una consulta i rebre la resposta a traves d'un canal anònim. Un exemple d'aquesta implementació és el protocol Tor [11]. El principal problema d'aquest aproximació és el temps que es consumeix en el procés i per tant l'eficiència és poca. Altres protocols que es focalitzen en aquest nivell de privacitat són els protocols PIR (Private Information Retrival), que permeten a un usuari rebre informació d'un servidor en possessió d'una base de dades sense revelar quin ítem s'intente rebre[12]. En el protocols PIR es requereix que hi hagi una col·laboració entre els WSE i els usuaris.

En el quart nivell de privacitat -sense informació personal- la identitat de l'usuari i la informació que demana no està disponible pel WSE. De totes maneres el cost computacional i d'ampla de banda requerits ho fan molt poc usables en la pràctica.

Cal dir, que hi ha mecanismes senzills per aconseguir un cert nivell d'anonimat en fer les consultes al WSE que inclouen: l'ús de *proxis* intermedis entre els usuaris i el WSE, o l'ús de la IP dinàmiques (per exemple, mitjançant DHCP). En el primer car, l'ús de proxis només mou l'amenaça de privacitat del WSE cap al *proxy*. Un *proxy* evitarà que el WSE perfili el usuaris, però el proxy si serà capaç de fer-ho. En el segon cas, l'ús de una IP dinàmica tampoc soluciona aquest problema ja que la política de renovació de la IP no la controla l'usuari i pot donar la mateixa IP a la mateixa adreça MAC.

1.1 Objectius del projecte

En el present treball ens centrarem en el nivell de privacitat de grup, utilitzant un protocol UUP. Concretament farem la implementació del protocol descrit a la referència *"Using social networks to distort users' profiles generated by web search engines", Computer Networks 54 (2010) 1343–1357 d'Alexandre Viejo i Jordi Castellà-Roca*[13] . Aquest sistema proporciona un mecanisme per a distorsionar els perfils d'usuari utilitzant un nivell de privacitat de grup. El grup estarà format per distints usuaris les xarxes socials, directament connectats entre ells o a traves d'altres usuari, i podran publicar i compartir informació.

Al protocol que implementarem es basarà en una xarxa col·laboradora totalment descentralitzada, on cada usuari és capaç de generar consultes que pot enviar-les directament al WSE o bé les pot re-dirigir cap als seus nodes veïnats. Per la seva banda, el node veïnat que rep la consulta té a la vegada la possibilitat de enviar-la cap directament al WSE o bé passar-la als seus respectius veïnats. Així una consulta serà re-enviada cap als nodes veïnats fins que un d'ells la envii al WSE. D'aquesta manera el WSE no serà capaç de perfilar un usuari en concret de la xarxa ja que no podrà fer el lligam consulta□usuari. De totes maneres, si bé no podrà perfilar un usuari en concret, si podrà "perfilar" la nostra xarxa social.

Especialment ens centrarem en un dels problemes que s'ens presenta a l'hora d'implementar aquest protocol i és el fet que un usuari en concret no coneix la tipologia ni l'estructura de la xarxa per a decidir cap a quin veïnat enviar la consulta. No coneix el nombre de connexions d'un usuari veïnat seu, solament coneix les connexions directes amb ell mateix. Per tant l'usuari haurà de fer una estimació d'aquesta estructura. En [13] s'en proposen quatre mètodes que fan aquesta tasca:

i. Assumir que tots els usuaris de la xarxa tenen el mateix nombre de connexions directes (veïnats)
ii. Utilitzant el nombre mitjà de consultes que l'usuari ha intentat enviar cap el veïnat que vol estimar.
iii. Utilitzant el nombre de consultes que cada veïnat de l'usuari ha intentat enviar-li. La idea és que un usuari amb molts connexions envia menys consultes a cada veïnat (distribució de les consultes).
iv. Utilitza també el nombre de consultes que cada veïnat de l'usuari ha intentat enviar-li. En aquest cas però, la idea que es vol captar és que un usuari amb moltes connexions acceptarà més consultes dels seus veïnats i per tant també re-dirigirà més cap als seus nodes.

L'eficiència d'aquests mètode presenten una clara mancança ja que, segons les simulacions dutes a terme a [13], el millor resultat s'obté amb el tercer mètode, però amb una probabilitat d'èxit relativament baixa, del 14,31% i amb una mitjana d'error de l'estimació de 2,12

Per tant en el present PFC ens proposam millorar aquest percentatge de probabilitat d'èxit. Destinats a tal fi, el projecte tendrà tres components diferenciades: per una banda s'implementarà i simularà el model proposat a [13] que ens servirà de base per a posteriors desenvolupaments. Per altra banda es proposaran aquestes millores en els mètodes d'estimació de la xarxa. I finalment es simularà el comportament amb aquests nous mètodes.

Per tant, els objectius que es volen assolir en aquest PFC presenten tan components d'implementació, investigació i simulació, i es poden resumir en els següents punts:

- Implementació del simulador del protocol descrit a [13].
- Avaluació del funcionament del simulador i comparació de les dades amb les publicades a [13]
- Proposta de nous mètodes per a fer l'estimació de la xarxa
- Avaluació del nou mètode proposat.

1.2 Justificació del projecte

La simulació dels mètodes de predicció i estimació de l'estructura de la xarxa proposats a [13] mostren que la seva eficiència es baixa. Ja s'ha dit abans que el millor mètode dels proposats té un

percentatge de probabilitat d'èxit del 14,31% amb una mitjana d'error de l'estimació de 2,12 (diferències entre el nombre real de veïnats i el valor estimat), que és relativament baix

El present treball vol contribuir a millorar aquest percentatge d'èxit, proposant i simulant nous mètodes d'estimació del nombre de veïns d'un usuari en concret. Amb aquesta millora es millorarà també la funció que estima el nivell d'exposició del perfil d'un usuari en concret. Aquest funció és molt important en el nostre model ja que ens retorna l'usuari més òptim que hauria de fer la consulta cap al WSE

Cal dir també que aquesta millora de l'estimació de l'estructura d'una xarxa social, no solament afecta a la implementació d'aquest protocol proposat a [13] sinó que es pot incorporar a l'estimació de qualsevol xarxa social descentralitzada.

1.3 Organització de la memòria

A la secció 2 es mostrarà l'estat actual d'estudi d'aquest escenari, juntament en les propostes que hi ha en les diferents organitzacions. A la secció 3 s'explicarà en detall el funcionament del protocol i mètodes utilitzats en aquest per a fer l'estimació de la tipologia de la xarxa. A la secció 4 descriurem els nou mètodes proposat per a millorar el rendiment d'aquesta estimació. A la secció 5 avaluarem en rendiment del nou esquema i el compararem amb els anteriors. En aquesta secció també descriurem el simulador del protocol desenvolupat. Finalment la secció 6 reportarà les conclusions finals d'aquest estudi i es discutirà la validesa d'aquests models desplegats en entorns reals.

2 Descripció de l'esquema

En aquest document intentam explicar en detall la referència sobre la qual ens basam per implementar el nostre protocol i les millores descrites en els objectius (Secció 1). Es podrà veure que a l'hora d'explicar l'esquema a implementar s'ha anat seguin l'estructura que s'utilitza a [13]

2.1 Esquema a implementar.

Com ja hem dita abans a la Secció 1, el protocol descrit a la referència ***"Using social networks to distort users' profiles generated by web search engines"***, Computer Networks 54 (2010) 1343–1357 d'Alexandre Viejo i Jordi Castellà-Roca[13] es centre en el nivell de privacitat de grup per a intentar distorsionar els perfils dels usuaris concrets que el WSE realitza. Per fer-ho necessita que els usuaris del grup estiguin organitzats en una xarxa social, entesa com una comunitat d'usuaris on cada un d'ells pot publicar i compartit informació i serveis (dades personals, blogs, i recursos en general)[2]. En algunes xarxes socials, els usuaris poden especificar el grau de confiança amb altres usuaris, assignant-los un nivell de confiança [3,4]. També és possible establir diferents tipus de relació entre els usuaris. En el cas del nostre protocol, no es requereix que la xarxa tengui cap node central (xarxa no estructurada), solament es necessita que els usuaris estiguin directament connectats entre ells o a través d'altres usuaris intermediaris, on cada usuari només coneix les seves pròpies connexions. La xarxa social que farem servir, descrita en el protocol[13], utilitza nivells de confiança associats a les connexions entre els usuaris.

En aquest esquema hi haurà present dues entitats:

- Usuaris: entitat que envia les consultes al WSE. S'organitza amb altres usuaris, creant la xarxa social i tenen un interès alt en salvaguardar la seva privacitat.
- WSE: entitat que executa la consulta que li arriba i retorna la resposta. No té cap interes en preservar la privacitat dels usuaris dels que rep la consulta.

Per tal de preservar la privacitat dels usuaris de la nostra xarxa social, enfront a perfilacions dels WSE, el protocol descrit intentarà ofuscar la relació consulta-usuari. En aquest sentit un usuari concret U, té la capacitat de generar consultes i enviar-les directament al WSE o bé passar-les als seus usuaris veïnats de la xarxa social. El veïnat quan rep una consulta també té la capacitat d'enviar-la directament al WSE o passar-la als seus propis veïnats. D'aquesta manera una consulta pot ser re-enviada als veïnats fins que un l'envia al WSE. Si totes les consultes generades en la nostra xarxa social són distribuïdes uniformement a la xarxa en perfil de cada usuari queda distorsionat amb les consultes que no li pertanyen. Però hi ha també problemes (apart del WSE) derivats d'un mal ús o abús del sistema, per part dels usuaris. Un d'aquests casos seria la no

cooperació.

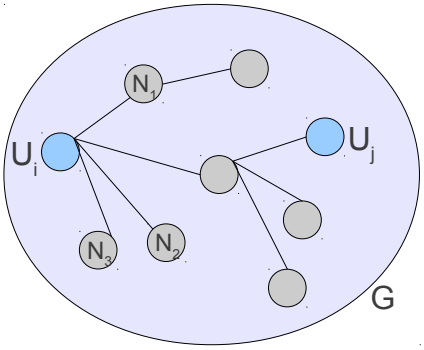
En el model anterior hi podem veure clarament tres adversaris:

- WSE: aquesta entitat és el principal adversari i enfront al qual, hem de garantir la privacitat. Hi ha informació que no podrem evitar enviar cap al WSE com pot ser la IP, les *cookies*, etc.
- Usuaris deshonests: un usuari amb intensions deshonestes podria utilitzar la seva pròpia funció per a perfilar altres usuaris ja que sap perfectament qui li envia la consulta.
- Usuaris egoistes: és aquell que utilitza els altres usuaris per enviar les seves consultes i no accepta cap consulta dels altres. Aquest comportament impedeix que l'enviament de les consultes es distribueixi uniformement a la xarxa i pot posar en perill la privacitat dels usuaris honests que fan un del protocol.

Per a poder dur a terme la distribució de consultes a tots els usuaris de la nostra xarxa necessitam utilitzar dues funcions que estimen respectivament el nivell d'exposició del perfil (funció ψ) i el nivell d'egoisme dels usuaris del sistema (funció ϒ).

2.2 Passos del protocol per enviar anònimament consultes al WSE

Assumim que un usuari Ui amb k veïnats (N1,...,Nk) de la nostra xarxa G vol enviar una consulta q al WSE.



Pas 1: Ui ha de decidir cap a quin veïnat re-enviar la consulta o bé executar-la ell directament. Aquesta decisió es fa depenent del nivell actual de privacitat de Ui. Sigui la funció ψ (Ui, N1,...,Nk) la funció que estima el nivell d'exposició de l'usuari Ui respecta als seus veïnats. Aquesta funció retorna l'usuari que hauria d'executar la consulta q cap al WSE. Si la funció ψ decideix que l'ha d'executar el propi Ui, aquest ho envia al WSE i el protocol finalitza aquí. D'altra manera, Ui envia la consulta cap a l'usuari indicat per ψ.

Pas 2:Assumim que ψ estima que és el veïnat Ni el que hauria d'executar la consulta q de l'usuari

Ui. D'aquesta manera Ui reenvia q cap a Ni. Per la seva banda, Ni accepta o rebutja q depenent del nivell d'egoisme de Ui respecta a Ni. Sigui □(Ni, Ui) la funció que calcula el nivell d'egoisme que Ni atorga a Ui. Aquesta funció retorna quan s'ha d'acceptar o no la consulta q.

- Si Ni accepta q, ha de repetir els mateixos passos que ha seguit Ui per a decidir que ha de fer amb q.
- Si Ni rebutja q, Ui ha de decidir, utilitzant la funció ψ qui dels restants veïnats candidats (tots els que no ha demanat anteriorment) hauria d'executar q, fins que algú accepta q.
- En cas que tots els veïnats de Ui,N1,..,Nk rebutgen la consulta, és el propi Ui qui fa l'enviament cap al WSE.

Pas 3: Assumim que un usuari Uj accepta una consulta q que ha generat un altre usuari Ui. Un com Uj ha obtingut la resposta a del WSE, l'ha de reenviar cap a Ui (que podria ser veïnat directe, o bé veïnat d'un veïnat directe,...). Aquest procés es repeteix fins que a arriba a l'usuari Ui, que ha generat la consulta q.

2.2.1 Funció ψ: estimació del nivell d'exposició del perfil

Considerem un usuari U que genera una consulta q i disposa d'una llista de veïnats directament connectats a ell (N1,..,Nk). U utilitzarà la funció ψ per a decidir qui dels conjunt d'usuaris {U,N1,..,Nk} hauria d'enviar la consulta al WSE. La funció ψ intentarà distribuir uniformement totes les consultes entre els usuaris de {U,N1,..,Nk}. Per fer això, ψ estima el nombre de consultes generades per U que cada veïnat ha probablement enviar al WSE. Sigui αi l'estimació del nombre de consultes de U que ha enviat el veïnat Ni. Es pot veure que ai és una estimació ja que U no pot saber si la consulta l'ha enviada directament Ni o bé l'ha reenviada cap as seus respectius veïnats. U sap cert el nombre de consultes que el veïnat Ni ha acceptat (τi). D'aquesta ψ computa αi de la següent manera:

αi = τi / nombre de veïnats de Ni

L'expressió anterior requereix el nombre de veïnats de Ni. Aquesta informació l'usuari U no la pot saber directament, sinó que l'ha d'estimar. A [1] s'han introduït quatre mètodes per a fer aquesta estimació. A la Secció 4 s'en proposaran nous mètodes que milloren el rendiments d'aquests quatre. Sigui □i aquesta estimació:

- **Mètode 1**: S'assumeix que tots els usuaris de la nostra xarxa social té el mateix nombre de connexions directes. Seguint aquest mètode, cada veïnat de U tendrà exactament el mateix nombre de connexions directes que U.
- **Mètode 2**: Sigui θ i el nombre de consultes que U ha intentat re-enviar cap al veïnat Ni. U també sap el nombre de connexions directes que té (c) i el nombre de consultes que el veïnat Ni ha intentat enviar cap a ell (ti). D'aquesta manera:

$$\beta_i = \frac{t_i \cdot c}{\theta_i}$$

- **Mètode 3**: Aquest mètode utilitza el nombre de consultes ti que cada veïnat ha intentat re-enviar a U. Cada un dels veïnats es posat a una llista ordenada respecta a ti. El veïnat amb un nombre ti més alt s'assumeix que només té una connexió. El segon veïnat de la llista s'assumeix que té dues connexions. D'aquesta manera aquest mètode fa servir la idea que un usuari amb distintes connexions envia menys consultes a cada veïnat.
- **Mètode 4**: És similar a l'anterior, però en aquest cas el veïnat amb menor ti s'assumeix que només té una connexió. Un usuari amb varies connexions acceptarà més consultes dels altres veïnats i com a resultat d'això, re-enviarà més consultes a cada veïnat.

2.2.2 Funció ϒ: estimació del nivell d'egoisme.

Per acceptar o rebutjar les consultes que els usuaris reben necessitam saber el nivell d'egoisme entre ells. Sigui U un usuari que rep una consulta q del seu veïnat N. U utilitza la funció ϒ per decidir si ha de acceptar o rebutjar q. Com ja s'ha dit, el propòsit de ϒ és penalitzar els usuaris que actuïn de manera egoista.

Inicialment l'usuari U assigna a cada veïnat una probabilitat p=1.0 (100% de probabilitat) d'acceptar consultes del veïnat. Suposem que U rep una consulta q d'un veïnat N. U accepta q amb una probabilitat $p_{U,N}$:

- Si U accepta q, cal fer les següents passes:
  - N incrementa la seva probabilitat d'acceptar consultes de U: $p_{N,U} = p_{N,U} + 2\gamma$, on γ és una constant definida al sistema
  - U disminueix la seva probabilitat d'acceptar consultes de N: $p_{U,N} = p_{U,N} - \gamma$

Aquests dos passos indueixen a acceptar consultes dels veïnats ja que poden caure en la situació d'una probabilitat zero, i queden aïllats. Per tornar a entrar al sistema hauran d'acceptar consultes dels seus veïnats.

- Si U rebutja q, aleshores N disminueix la seva pròpia probabilitat d'acceptar consultes de U: $p_{N,U} = p_{U,N} - \gamma$

2.3 Anonimat dels usuaris

En l'esquema descrit a [1] l'anonimat dels usuaris s'aconsegueix entre usuaris que no estan directament connectats. És a dir un usuari que rep una consulta sap exactament qui li envia la petició, però no l'usuari que ha generat la consulta, ni l'usuari que que l'ha reenviada cap al veïnat que ha fet la petició. L'existència d'aquest anonimat pot fer que usuaris deshonests vulguen enviar

consultes il·legals. D'aquesta manera, l'usuari que envia la consulta no seria capaç de demostrar que la consulta no l'ha generada ell sinó que li ve d'un altre usuari de la xarxa. En canvi, l'usuari generador de la consulta obtendria la resposta de la consulta, sense haver fet cap infracció. El protocol descrit utilitza un mecanisme contra això anomenat a posteriori i no entra en filtrar les consultes per a detectar les consultes legals d'aquelles que no ho son.

El protocol que utilitzam utilitza un mecanisme de responsabilitat per a provar la innocència dels usuaris intermediaris involucrats en la cerca, que actuen de manera honesta i proporciona un mecanisme per a demostrar a terceres parts (autoritats governamentals) que no són els responsables de la generació de la consulta.

Per assegurar això es fa servir ens certificats en cada transacció entre dos usuaris. Els certificats de cada transacció entre dos usuaris està format per la identificació dels dos usuaris involucrats en la transacció, la consulta re-enviada, i un *timestamp* que identifica el moment en que es va fer. Cada usuari que rep un certificat l'ha de guardar en un lloc segur, de tal manera que si és interrogat per una autoritat podrà demostrar que és únicament un node intermediari amb un cost computacional acceptable.

2.4 Valors de confiança

Com ja hem dit abans la xarxa social que farem servir, descrita en el protocol[1], utilitza un valor de confiança associat a les connexions entre els usuaris. Així l'usuari que genera la consulta pot inicialment decidir només re-enviar la consulta als seus veïnats amb un valor de confiança igual o major a un cert valor, sense executar la funció ψ. De manera semblant, un usuari pot rebutjar una consulta que prové d'un veïnat amb un nivell de confiança menor que un cert llindar, sense utilitzar la funció Υ.

2.5 Prevenció de la resposta

[1] ens proposa tres mecanismes per a fer impossible un canvi de la resposta correcta proporcionada per l'usuari que executa la consulta, per una resposta falsa.

- Integritat dels usuaris: En aquests tipus de xarxes socials amb nivells de confiança entre els usuaris, l'usuari que ha generat la consulta (propietari legítim) obté la resposta de la consulta i un valor de confiança, que es calculat utilitzant el valor de confiança de tots els usuaris (nodes) que intervenen per fer la consulta. Depenent d'aquest valor el propietari legítim pot acceptar o no la resposta.
- Mecanismes de responsabilitat: A [1] es proposa mecanismes de responsabilitat similars als descrits a la secció 2.2, per a detectar si hi ha hagut modificacions en la resposta. No evita la possibilitat d'un canvi malèvol a la resposta, sinó que legitima al propietari de la consulta a detectar una resposta falsa i cercar l'usuari responsable.

- Mètodes basats en agents mòbils: A [1] s'ens diu que hi ha tècniques utilitzades pels agents mòbils) per a traçar aquests problemes.

3 Disseny i implementació.

Marcàvem a la introducció que entre els objectius principals del projecte hi ha quatre punts bàsics a assolir:

- Implementació d'un simulador descrit a "Using social networks to distort users' profiles generated by web search engines", Computer Networks 54 (2010) 1343–1357 d'Alexandre Viejo i Jordi Castellà-Roca[13].
- Avaluació del funcionament del protocol i comparació de les dades.
- Proposta de nous mètodes d'estimació de la xarxa.
- Avaluació dels nous mètodes proposats.

En aquest apartat ens centrarem en el disseny i la implementació del simulador. Atès que el realment important pel projecte és l'anàlisi de les dades proporcionades pel simulador, s'ha optat per dissenyar un simulador molt bàsic però suficient per el desenvolupament del protocol descrit.

En els següents punts es donarà una visió del disseny que es vol dur a terme. Primerament es plantejaran els requisits que hauria de tenir el simulador. Els altres apartats es centren en el disseny de l'aplicació. En els pròxims apartats es dóna una visió de l'arquitectura global de l'aplicació, tan a nivell lògic com a nivell físic, així com dels distints *frameworks* utilitzats i provats per a realitzar-ho.

3.1 Característiques del simulador.

Abans d'entrar al disseny pròpiament dit i a la seva implementació cal parlar de les característiques pròpies del simuladors (ens hem imposat aquestes característiques com a requisits del simulador). Degut a la natura del projecte el simulador ha de presentar les següents propietats:

- **Robust**: El sistema ha de ser tolerant a determinades errades. El sistema ha de detectar quan la simulació a un determinat node està entrant a un bucle infinit i continuar amb les simulacions dels altres nodes sense aturar els fils d'execució.
- **Ràpid**: Cal que el simulador sigui prudent amb els accessos a disc o a base de dades i sigui el més ràpid i òptim possible degut a la gran quantitat d'informació que ha de generar.
- **Escalable**: S'ha d'aconseguir un sistema mínim escalable on es puguin executar diverses simulacions (processos diferents) a l'hora. També cal poder executar el simulador des de diverses màquines i recollir la informació de manera centralitza.
- **Usable**: Ha de permetre a l'usuari canviar els paràmetre de l'execució sense haver de compilar cada pic. També cal presentar els resultats de manera òptima per a la interpretació de la simulació.
- **Integrable**: s'hauria de dissenyar el simulador de tal manera que fos fàcilment integrable en

altres sistemes i per això cal que el codi:

- Estigui ben estructurat (divisió per capes)
- Re-usable

3.2 Requisits del simulador i actors del sistema

A continuació mostrarem els requisits que ha de tenir el nostre simulador bàsic:

- Centralització de dades: s'ha d'aconseguir que totes les dades de les diferents simulacions (de diferents màquines, si és possible) s'emmagatzemin centralitzades a un magatzem per a després analitzar-les.
- Càrrega de xarxes: el simulador ha de ser capaç de carregar xarxes al sistema a partir de fitxers de col·leccions de dades de *Stanford Large Network Dataset Collection* i/o altres tipus de col·leccions de dades.
- Establir els següents paràmetres abans de cada simulació:
  - Mètode d'estimació amb el que volem simular.
  - Nombre d'usuaris màxim a carregar.
  - Factor lambda que per defecte serà sempre 0,02.
  - Probabilitat mínima amb la que acceptam consultes (per defecte 0.90)
  - Nombre de consultes a generar
  - Node/Usuari que genera les consultes. Si no s'indica vol significa que són tots els usuaris de la xarxa els que generen totes les consultes
- Dades a emmagatzemar de cada simulació: S'han de guardar dades de dos tipus: referents a les dades globals de la simulació i de cada simulació dades de cada node
  - Dades globals de la simulació:
    - Data en que s'ha feta la simulació (hora, minut i segons)
    - Xarxa simulada.
    - Mètode simulat.
    - Usuaris de la xarxa simulada
    - %Encerts mitjà de la xarxa
    - Desviació típica.
  - Dades de cada node de la xarxa
    - ID del node.
    - ID del node veïnat.
    - Nombre de consultes rebudes.

        - Nombre de consultes enviades.
        - Nombre de consultes acceptades pel veïnat.
        - Nombre de consultes acceptades del veïnat.
        - Nombre de intents i encerts a l'hora d'estimar el nombre de nodes del veïnat.
        - Temps mitjà en rebre les consultes al veïnat.
        - Temps mitjà en enviar les consultes al veïnat.
    - Recuperar els resultats de les simulacions guardades.
    - Capacitat de mostrar per a cada node de la xarxa simulada les dades resultants.
    - Capacitat de mostrar els resultats globals de la xarxa simulada: %encerts d'encerts, desviació típica dels nodes, temps mitjà per node.

Bàsicament, els **actors** que intervenen al sistema és un:

- Simulador: és la persona encarregada de carregar la xarxa que es vol analitzar, fixar els paràmetres de configuració, executar la simulació i analitzar les dades retornades.

3.3 Diagrama d'entitats

En aquest apartat analitzarem el model d'entitats utilitzat a l'hora d'implementar el simulador. A la figura següent es pot veure aquest diagrama de model. En general s'ha intentat aconseguir els següents objectius a l'hora de dissenyar el model d'entitats:

- **Flexibilitat**, especialment per aquelles parts que més han canviat al llarg del desenvolupament del projecte (mètode d'estimació, usuari i xarxa)
- **Agrupació** en paquets de les entitats segons la capa estructural a la qual pertany (veure següent secció). Així tenim que hi haurà classes completament destinades a la presentació (SimuladorGràfic, Test), entitas destinades a la gestió de l'emmagatzemament dels resultats i les simulacions (SimuladorDao, Simulacio, HibernateUtilities,...) i d'altres que encaixen en el procés de negoci del protocol (Xarxa, User, MetodeEstimacio,...)
- **Senzillesa**: ja que el nucli del projecte no es tan la implementació del simulador com l'obtenció de dades i el seu anàlisi, s'ha intentat fer el disseny el més senzill possible.

A continuació entrarem en detall per a cada un dels elements i entitats que formen part d'aquest disseny. Ho anirem analitzar agrupant les entitats segons a la capa de l'arquitectura per nivells a la qual pertany.

a) **Presentació**:
   - *SimuladorGrafic*: És una classe de Java-Swing encarregada de ser la pantalla principal de l'entorn de simulació gràfic. Hereta les propietats de Jframe i implementa el mètode de la interfície Simulador anomenat simula().
   - *Test*: És una entitat de tipus *main* de Java que representa un simulador més simple que el presentat pel SimuladorGrafic. També implementa el mètode simula() de la interfície Simulador. Aquesta entitat presenta per pantalla (sortida estàndard) els resultats de les simulacions, que a base d'ajustar els paràmetres d'entrada que es demana a l'usuari, executa una simulació.

b) **Negoci o servei:**
   - *Xarxa*: És la entitat que ens abstreu totes les propietats que presenta una xarxa social capaç d'executar el protocol descrit a [13]. Bàsicament es composa d'una col·lecció d'usuaris que interaccionen entre ells segons el protocol descrit. Proporciona els mecanismes necessàries per a poder carregar la xarxa a partir d'un fitxer de dades, fixar els paràmetres de l'execució, executar de manera massiva consultes cap als usuaris i obtenir-ne els resultats. Entre els principals mètode de servei podem comentar els

següents:

- carregaXarxa: mètode encarregat de afegir tots els nodes a partir d'un fitxer d'entrada.
- afegeixRelacio: mètode que afegeix una relació de veinat entre dos usuaris. Si els usuaris no existeixen a la xarxa els dóna d'alta
- generaConsulta: mètode que fa que l'usuari indicat generi una consulta al sistema.
- generaConsultes: mètode que fa generar a tots els usuaris o bé a una llista de nodes en concret de la xarxa, un nombre de consultes inidicat. Aquest mètode és el que s'utilitza a l'hora de generar consultes massivament.
- obtenirLogs: mètode per obtenir els resultats d'un node en concret, després de l'execució de la simulació. Retorna una estructura de mapa on les claus són els veïnats del node que estam inspeccionant i el seu valor és la informació associada a aquest veïnat (és a dir, un objecte de tipus InformacioVeinat).
- obtenirDadesTempsAgrupadesPerVeinat: mètode que ens retorna una estructura de mapa on les claus són el nombre de veïnats i els valors representen el temps mitjà en rebre consultes d'un node que tè un nombre de veïnats igual a la clau especificada.
- obtenirDadesRendiment: mètode mitjançant el qual obtenim les dades del rendiment global de la simulació.
- carregaRegressio: mètode mitjançant el qual obtenim de la base de dades la regressió lineal de la xarxa segons el nombre d'usuaris, la xarxa que analitzam, les consultes generades i el mètode.

- *User*: És la entitat que emula el comportament 'un node/usuari de la xarxa social que actúa segons el protocol descrit al llarg del projecte. És l'encarregat de guardar en memòria els resultats que va obtenint el propi node al llarg de l'execució. Cada node està compost per una col·lecció de nodes veïnats amb una informació associat a cada un d'ells. Cada cop que es produeix un esdeveniment (rebem una consulta d'un veïnat, enviam una consulta a un node veí, acceptam una consulta, etc.) s'actualitza aquesta informació. Bàsicament és aquesta entitat el que implementa el protocol ja que cada node ha de ser autònom per si mateix i només és ell que ha de decidir cap a on "encaminar" la consulta o si acceptar o no consultes que li arriben.
- *UtilQuery*: entitat auxiliar que ajuda als nodes a calcular l'usuari veïnat amb menor nivell d'exposició, i a actualitzar les probabilitats després de que un node hagi acceptat o rebutjat una consulta.

- *MetodeEstimacio*: És la interfície que exposa la funció que qualsevol mètode d'estimació ha d'implementar.
  - *MetodeSimple*: Es correspon al mètode 1 dels llistats a [13]. En aquest mètode s'assumeix que tots els usuaris de la nostra xarxa social té el mateix nombre de connexions directes. Seguint aquest mètode, cada veïnat de U tendrà exactament el mateix nombre de connexions directes que U.
  - *MetodeNombreMitjaConsultes*: És el mètode 2 de les especificacions del protocol.Sigui θ_i el nombre de consultes que un node U ha intentat re-enviar cap al veïnat N_i. U també sap el nombre de connexions directes que té (c) i el nombre de consultes que el veïnat N_i ha intentat enviar cap a ell (t_i). D'aquesta manera:

    $$\beta_i = \frac{t_i \cdot c}{\theta_i}$$

  - *MetodeAltConsultesEnviades*: És el mètode 3 del protocol. Aquest mètode utilitza el nombre de consultes t_i que cada veïnat ha intentat re-enviar a U. Cada un dels veïnats es posat a una llista ordenada respecta a t_i. El veïnat amb un nombre t_i més alt s'assumeix que només té una connexió. El segon veïnat de la llista s'assumeix que té dues connexions. Aquest mètode utilitza una *inner* classe de tipus **comparador** i un mapa ordenat (TreeMap) per a ordenat de manera creixent els veïnats d'un node, segons el nomdre de consultes rebudes d'aquest veïnat. Amb això asseguram que la complexitat de l'algorisme sigui de l'ordre de **log(n)**.
  - *MetodeBaixConsultesEnviades*: És el metode 4 del protocol.És similar a l'anterior, però en aquest cas el veïnat amb menor t_i s'assumeix que només té una connexió. Aquest mètode també utilitza una *inner* classe de tipus **comparador** i un mapa ordenat (TreeMap) per a ordenat de manera decreixent els veïnats d'un node, segons el nomdre de consultes rebudes d'aquest veïnat. Asseguram que la complexitat de l'algorisme sigui de l'ordre de **log(n)**.
  - *MetodeIntervalTemps*: És el mètode nou proposat arran de les simulacions obingudes per tal de millorar el rendiment dels altres. Aquest mètode utilitza uns aproximació al comportament de la xarxa mitjançant una regressió lineal que a partir del temps mitjà en rebre una consulta del veinat per a aproximat el nombre de veïnats que té. Presenta també una complexitat de cost **log(n)**.
  - *MetodeTempsMitjaAlt*: Eś un mètode nou també proposat (veure properes seccions). Es base en el mètode 4 i s'assumeix que el veïnat amb menor temps mitjà d'enviament de consultes només té una connexió. El segon, dues connexions, i així

successivament. Aquest algorisme també presenta una complexitats de cost log(n).

c) Accés a dades:

- *HibernateUtilities*: classe amb mètode estàtics que ens retorna una factoria de sessions hibernate per a accedir a la base de dades.
- *SimuladorDao*: interfície amb els mètodes per a poder enregistrar i recuperar la informació de la base de dades.
- *Simulacio*: Entitat POJO que es mapeja cap a la taula SIMULACIO de la base de dades i en representa una simulació
- *Resultat*: Entitat POJO que també es mapeja cap a la taula RESULTAT de la base de dades i guarda informació dels resultats de cada node de la simulació

Diagrama d'entitats:

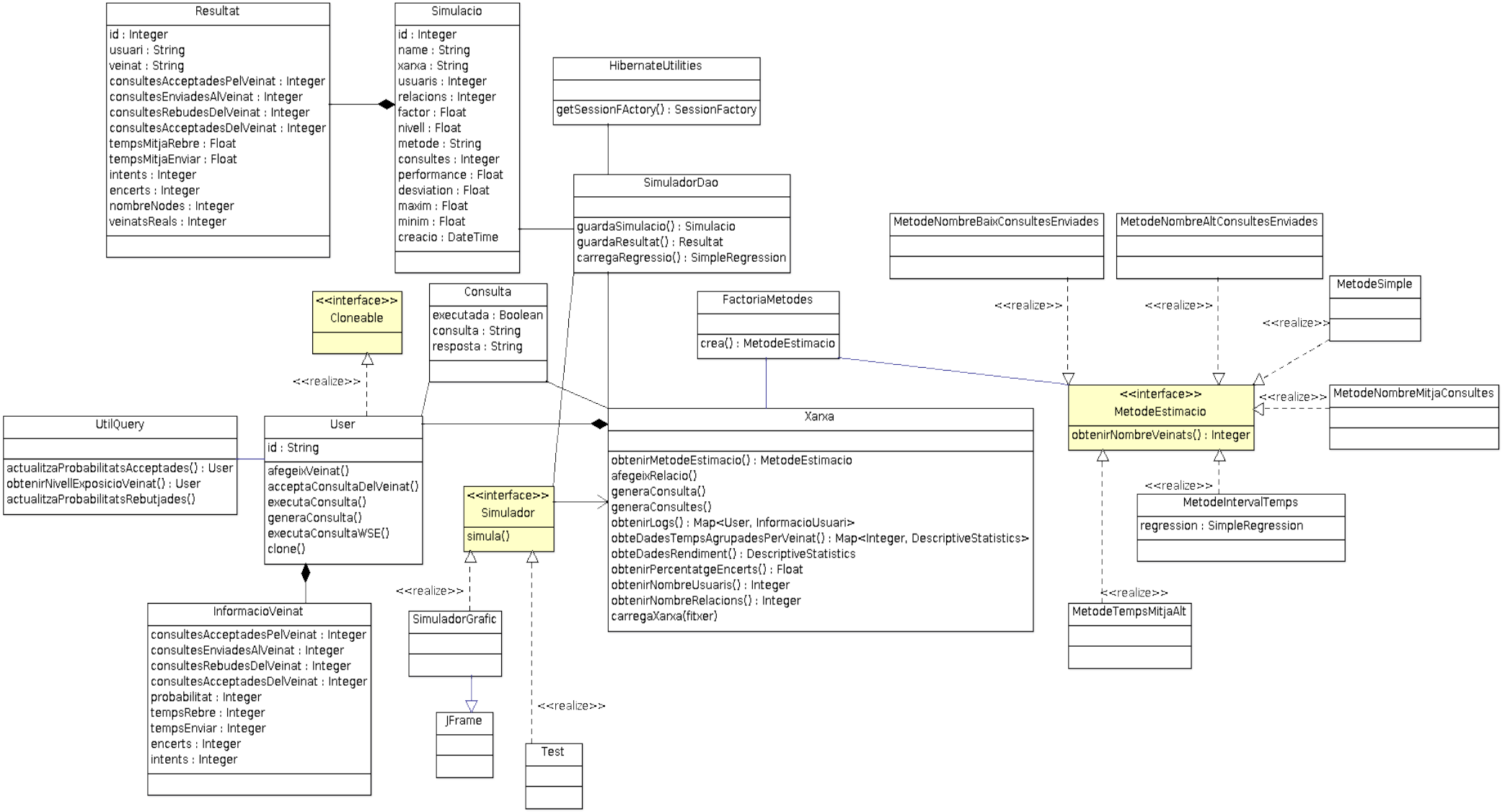
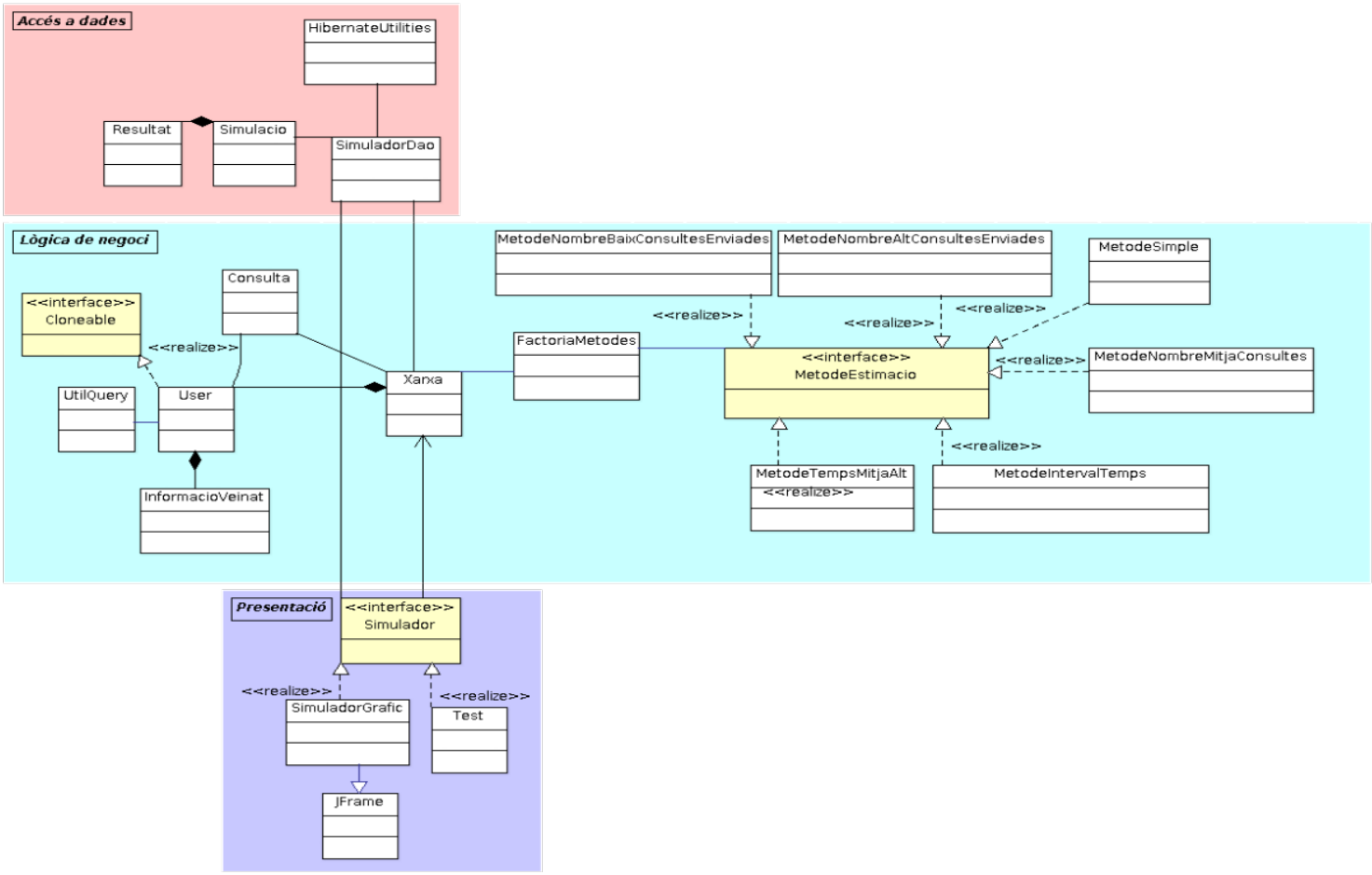


Diagrama d'entitats agrupats:



3.4 Disseny de la base de dades

Als apartats anteriors s'ha comentat la necessitat de guardar les dades de les simulacions per analitzar-les després. Deguat a la gran quantitat de dades generades i a la possibilitat de gestionar-les ràpidament i de manera òptima, s'ha decidir emmagatzemar-les en una base de dades relacional. S'ha escollit com a motor de base de dades MySql, encara que és facilment transportable a qualsevol altra base de dades relacional.

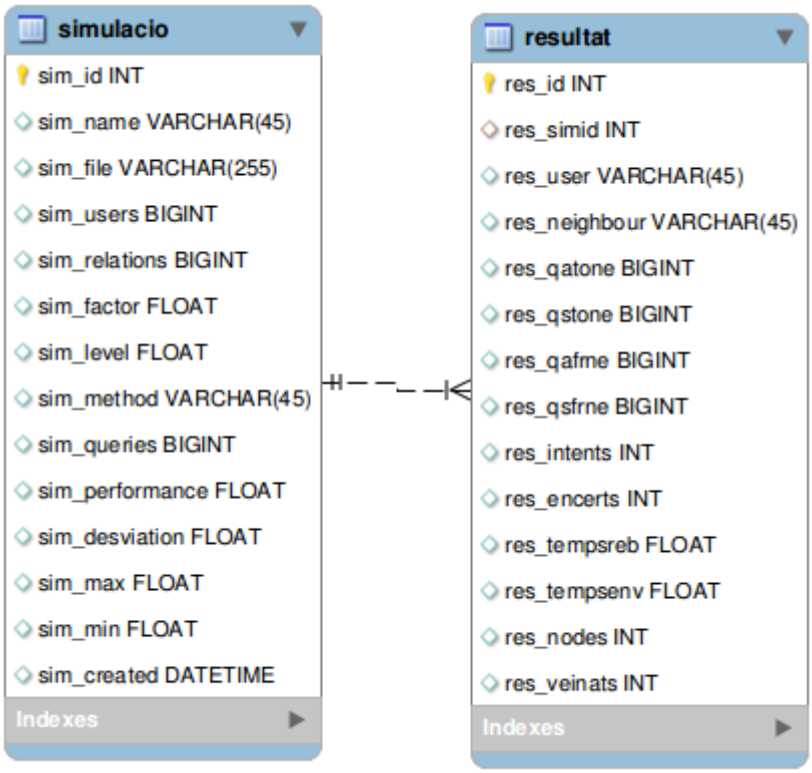
L'estructura de la base de dades per donar solució a les necessitats especificades als requeriments és molt simple i consisteix en les següents taules:

- Taula de simulacions (SIMULACIO): Són les dades globals de cada simulació i presenta els següents camps:
  - Id (sim_id): identificació numèrica del registre de simulació. És un camp auto-numèric que identifica unívocament el registre.
  - Nom (sim_nom): nom de la simulació.
  - Xarxa (sim_file): nom del fitxer que configura la nostra xarxa.

- Usuaris (sim_users): nombre d'usuaris que conformen la nostra xarxa social simulada.
- Relacions (sim_relations): nombre total de relacions creades en la simulació.
- Mètode (sim_method): mètode d'estimació de la mida de la xarxa utilitzat en aquesta simulació.
- Consultes (sim_queries): nombre de consultes generades en aquesta simulació.
- Factor (sim_factor): factor lambda utilitzat en la simulació.
- Nivell (sim_level): nivell de probabilitat mínim per a que els nodes acceptin consultes del seus nodes veïnats.
- %Encerts (sim_performance): mitjana de percentatge d'encerts dels nodes de la xarxa, en aquesta simulació.
- Desviació (sim_desviation): és la desviació estàndard del % d'encerts.
- Màxim (sim_max): % d'encerts màxim aconseguits en aquesta simulació.
- Mínim (sim_min): % d'encerts mínim aconseguits en la simulació.
- Data de creació (sim_created): data en format *timestamp,* en que es va fer aquesta simulació.

- Taula de resultats de simulació (RESULTAT): En aquesta taula s'emmagatzemarà els resultats de cada node que participa en la simulació. Cal tenir en compte a l'hora de crear-la que possiblement creixerà molt (de l'ordre de milions de registres). Presenta els següent camps :
  - Id (res_id): identificació numèrica del registre de resultat. És un camp auto-numèric que identifica unívocament el registre.
  - Simulació (res_simid): identificador de la simulació.
  - Identificador del node (res_user): cadena identificadora del node que aporta aquests resultats.
  - Identificador del veïnat (res_neighbour): cadena identificadora del node veïnat sobre el que es fan els resultats.
  - Consultes enviades al veïnat (res_qstone): nombre de consultes que el node identificat amb res_user envia cap a el seu veïnat identificat per res_neighbour.
  - Consultes acceptades pel veïnat(res_qatone): nombre de consultes que el node identificat amb res_user envia cap a el seu veïnat identificat per res_neighbour i són acceptades per aquest darrer.
  - Consultes enviades pel veïnat (res_qsfrne): nombre de consultes que el node identificat amb res_neighbour envia cap al node res_user.

- Consultes acceptades del veïnat(res_qafdrne): nombre de consultes que el node identificat amb res_user accepta del seu veïnat identificat per res_neighbour.
- Intents (res_intents): nombre d'intents que el node identificat per res_user ha intentat estimar la mida de veïnats de res_ neighbour.
- Encerts (res_encerts): nombre d'encerts que el node identificat per res_user ha estimar la mida de veïnats del node identificat per res_ neighbour correctament.
- Temps mitjà en rebre (res_tempsreb): temps mitjà en rebre consultes del node identificat per res_user provenides del node identificat per res_ neighbour.
- Temps mitjà en enviar (res_tempsenv): temps mitjà en enviar consultes del node identificat per res_user cap al node identificat per res_ neighbour.
- Nombre veïnats (res_nodes): nombre de veinats que el node identificat per res_user poseeix. Aquest camp si bé és redundant, ens facilita molt a l'hora de optimitzar les cerques i fer càlculs.
- Nombre veïnats reals (res_veinats): nombre real de veinats que el node identificat per res_ neighbour poseeix. Aquest camp si bé és redundant, ens facilita molt a l'hora de optimitzar les cerques i fer càlculs.



Quan als scripts de creació de les taules de base de dades s'ofereixen al projecte dins la carpeta pfc/scripts. Cal dir que quan es desplega l'aplicació per primera vegada les taules es creen automàticament. Igualment es poden trobar els scripts annexats al present document.

3.5 Estructura del projecte i llibreries

En aquesta secció es descriurà la infraestructura tecnològica que ha estat necessària per a la realització del projecte i s'enumeraran totes les eines que s'han utilitzat per a portar-lo a terme, tant a nivell de desenvolupament i proves com a nivell de redacció i maquetació de la present memòria. També es justificaran en aquests apartats les decisions sobre algunes qüestions de disseny.

3.5.1 Llenguatge de programació.

El llenguatge de programació utilitzat per al desenvolupament de tot el projecte ha estat Java SE [13], concretament la distribució Java Development Kit (JDK) 1.6.0.29 [14].

3.5.2 Llibreries addicionals.

Les funcionalitats oferides per la distribució 1.6.0.29 del JDK de Java s'han complementat amb una sèrie de llibreries "open-source" que han aportat funcionalitats específiques necessàries per a certs aspectes del desenvolupament. Aquestes llibreries són:

- **Apache Commons Math**: És una llibreria lleugera que conté components matemàtics i estadístics que aborden els problemes més comuns que no estan disponibles en la llibreries pròpies de la JDK. No té dependències amb cap llibreria externa ja que s'emfatitza en la senzillesa i la facilitat d'ús dels components integrats.
- **JUNG (Java Universal Network/Graph):** Són un conjunt de llibreries de programari Java, que proporcionen un llenguatge comú i extensible pel modelatge, anàlisi i visualització de dades que es poder representar en un graf o una xarxa. L'arquitectura de JUNG està dissenyada per donar suport a una varietat de representacions de entitats i les relacions, com ara gràfic dirigits i no dirigits, gràfic multi-modal, gràfic amb vores paral·lels i *hypergraphs*. Proporciona un mecanisme per a l'anotació dels gràfics, les entitats i les relacions amb les meta-dades. Això facilita la creació d'eines d'anàlisi de conjunts de dades complexes que es poden examinar les relacions entre entitats, així com les meta-dades adjunta a cada entitat i relació.
- **Hibernate**: Hibernate és una llibreria pel llenguatge Java destinada a fer mapeig objecte-relacional (ORM). Proporciona un marc per a l'assignació d'un model de domini orientat a objectes a una base de dades relacional tradicional. Hibernate resol el desajust objecte-relacional d'impedància mitjançant la substitució dels accessos directes de base de dades

amb funcions d'alt nivell de manipulació d'objectes.

- **MySql Connectors**: Com serà comentat més endavant, per a la implementació del magatzem de dades de les diferents simulacions del protocol s'ha fet servir un sistema gestor de bases de dades MySql. Per a poder establir connexions amb aquest tipus de base de dades des de les aplicacions Java és necessari l'ús dels *connectors* apropiats. En aquest cas s'ha fet servir la versió 5.1.12 dels *drivers* de MySql, incorporats dins el mateix projecte.
- **Ant**: Encara que no siguin llibreries del projecte cap dir que per a la construcció del mateix s'ha de disposar d'aquest bastiment,Una altra eina essencial per a la instal·lació i configuració del projecte és l'ant. Aquest utensili es pot baixar directament de http://ant.apache.org/ i s'utilitza per a realitzar tasques concretes i repetitives a l'hora de gestionar, construir i desplegar un projecte Java. Es pot instal·lar qualsevol versió superior o igual a 1.7, encara que amb versions més velles com la 1.6 també funcionaria.

3.5.3 Entorn de desenvolupament

L'entorn de desenvolupament del projecte utilitzat per a la implementació en llenguatge Java de tota la solució de programari ha estat IntelliJ IDEA en la seva versió "Community Edition". Aquest entorn de desenvolupament IntelliJ IDEA és un entorn integrat "open-source" que proporcionat moltes funcionalitats de cara al desenvolupament:

- Editor de codi intel·ligent que analitza el codi que es va desenvolupant amb refactoritzacions, inspecció de codi i propostes d'acció ràpides.
- Integració amb **Junit** i *TestNG* amb una interficie per a fer tests.
- Ús de **Maven** i **Ant** per a la construcció i instal·lació de projectes.
- Soport per a molts bastiments: Groovy, Hiberate, etc.

Cal dir però que el projecte no depèn de cap entorn de desenvolupament i s'ha fet de tal manera que es pugui compilar, generar i executar sense la participació de cap IDE, just a base de tasques Ant.

3.5.4 Estructura lògica del projecte

Encara que l'aplicació del simulador sigui una aplicació local (atacant a una base de dades relacional, que no té per que estar en la mateixa màquina), s'ha dissenyat pensant en una arquitectura en tres capes. S'entén per aplicació multi-capa (arquitectura amb n-capes) com una arquitectura de programari tipus client-servidor en la qual la presentació, gestió de dades i lògica de negoci o procés lògic, es troben separats de manera lògica (o inclús a vegades físicament, utilitzant

distintes màquines).

La separació de les capes es fa segons les responsabilitats o papers (rols) de cada una. Així distintes responsabilitats han de formar distintes capes (separació d'incumbències) que interaccionen entre elles mitjançant interfícies.

El model bàsic d'aplicacions n-capes, com hem dit abans, és el model 3-capes:

- **Presentació**: gestiona la navegabilitat i interacció amb l'usuari, és a dir aquells aspectes relacionats amb la lògica de presentació de l'aplicació.
- **Dades o persistència**: aquesta capa és l'encarregada de la persistència de les dades , és a dir gestiona l'emmagatzemament i accés a les mateixes.
- **Domini de l'aplicació**: resultat de l'anàlisi funcional de l'aplicació. Hi haurà les classes que ens abstreuen el model conceptual així com les operacions de negoci requerits.

Separant les responsabilitats dels processo de l'aplicació en diverses capes fa més fàcil **l'escalabilitat** de l'aplicació. Si consideram que cada capa pot estar en màquines físicament separades (no és el cas, en principi, del projecte del simulador) podem suportar millor càrregues de tràfic i connexions elevades.

Utilitzant aquest model d'arquitectura per capes, es permet separar el treball de manera més clara. Es pot anar enfocant el desenvolupament de l'aplicació de forma paral·lela de les tres parts: presentació (part client), persistència o gestió de dades i la lògica d'aplicació. Una aplicació estructurada en n-capes facilita el seu manteniment i la seva llegibilitat. I fa que els seus components siguin re-utilitzables.

El disseny estructural del simulador segueix un poc l'esquema anterior de tres capes. Passem a comentar les decisions de disseny preses a cada capa.

3.5.4.1 Capa de presentació

S'ha implementat dos accessos distints al simulador amb façanes distintes. Per una banda tenim una entrada simple a través de la línia de comanda, i per l'altre tenim un entrada gràfica mitjançant component de Java Swing.

- Linea de comandes. Al projecte s'ha implementat una simulació simple utilitzant la línia de comandes. És un client molt senzill que utilitza l'entrada estàndard per demanar els paràmetres de l'execució i per mostrar per la sortida estàndard els resultats d'aquesta simulació. S'ha optat per desenvolupar aquesta entrada al simulador per varis motius:
  - Ens ha servit per a testejar i provar funcionalitats sense executar cada cop un entorn gràfic

- Ens ha permès programar moltes execucions alhora (varis processos funcionant a la mateixa màquina) sense la necessitat d'interactuar amb la persona responsable.

- Simulador gràfic: client implementat en les llibreries de Java Swing i representa un *JFrame* on es carrega la xarxa a simular, permet simular i canviar diferents paràmetres i mostra els resultats del nodes en concret (de manera interactiva) i del conjunt de la xarxa. S'ha escollit implementar-ho d'aquesta manera pels següents motius:
  - Possibilitat de visualitzar l'estructura de la xarxa (és una bona opció per pocs usuaris, i per provar, sobretot al començament, el balanceig del protocol descrit)
  - Possibilitat de visualitzar amb interactivitat els resultats de certs nodes de la xarxa.
  - Permetre veure de manera intuïtiva possibles colls de botella a la xarxa.
  - No s'ha optat per solucions amb client web ja que uns dels objectius era realitzar un simulador bàsic que ens servís per provar el protocol i generar dades per avaluar els mètode d'estimació.
  - Les llibreries de Swing, al anar ja integrades dins les JDK de Java no s'han d'utilitzar llibreries extres.

3.5.4.2 Mecanismes de persistència i accés a dades.

Aquesta capa s'encarrega de l'accés a les dades i la conversió de l'estructura relacional pròpia de les bases de dades relacionals a les entitats del nostre domini. En aquesta capa hi té un paper fonamental dues peces: Hibernate i els DAO's.

- **Hibernate**: aquesta tecnologia s'encarrega del mapeig d'objectes relacionals (ORM[1]) amb les taules de la nostra base de dades relacionals. Per tant és pot dir que Hibernaun te estableix un lligam entre els nostres objectes de domini (POJO'S) amb les taules de la base de dades, mitjançant uns arxius de configuració (*.hbm.xml) on es defineix aquesta relació.
- **DAO (Data Access Object):** són unes classes o components centralitzats que subministren una interfície a les capa/capes que l'utilitzen per a l'accés a les dades.

NOTA: Realment, per a mida del projecte del simulador no era necessari incloure un altre bastiment com és Hibernate. Però pensant donat el seu coneixement anterior, la facilitat de canvi a l'hora de modificar algun entitat relacional, la facilitat de configuració de les connexions (ens abstreu del problemes típics de les aplicacions JDBC com puguin ser el tancament de connexions,etc.) ho varem incloure com a tecnologia al projecte.

1 El mapeig d'objectes relacional (ORM) és una tècnica de programació per convertir dades de llenguatges de programació orientats a objectes en la seva representació en bases de dades relacionals, a través de la definició de les correspondències entre els diferents sistemes

3.5.4.3 Lògica de negoci i serveis del simulador.

La lògica del projecte, és a dir l'implementació pròpiament dit del protocol descrit a la referència [1] s'ha dut a terme amb objectes de Java nadiu, sense l'ús de *sockets* ni de llibreries especialitzades en el tractament de xarxes d'usuaris p2p (JXTA, Pastry, etc.). No és l'objectiu del projecte simular escenaris on si es necessari l'ús d'aquestes tecnologies, per exemple si es vol simular el temps de propagació entre els nodes, o simular casos on alguns nodes no estan connectats. En el nostre cas, els nodes/usuaris de la xarxa estan sempre actius i disponibles per enviar i rebre consultes.

Tampoc no entra dins l'abast d'aquest projecte la simulació real d'enviaments de consultes cap al WSE. En el nostre cas es suposa que si un node ha d'enviar una consulta al WSE en tot moment ho podrà fer i no fallarà.

3.5.5 Estructura física del projecte

Fins ara s'ha explicat el model lògic de l'aplicació. Passarem ara a explicar de quina forma està organitzada físicament el projecte, indicant en cada cas a quina capa del model lògic es correspon. Com es pot veure en la figura anterior el projecte físic es divideix en distintes capes lògiques explicades anteriorment. Hi ha també una sèrie de directoris necessaris que intervenen en la configuració, compilació i desplegament del projecte. Passem a descriure que conté cada directori en qüestió:

- **etc**: aquí hi figura arxius de configuració globals de tota l'aplicació. Aquí hi ha present el mapeig ORM i la configuració de Hibernate.
- **lib**: directori on hi ha totes les llibreries necessàries per la compilació, construcció i desplegament correcta de l'aplicació. Bàsicament conté arxius de tipus .jar's que el projecte necessita, organitzats ens distintes carpetes
- **output**: és un directori dinàmic que es genera quan es construeix l'aplicació. Es tracta d'un directori temporal on es dipositen les classes compilades i els descriptors que es generen.
- **uml**: directori on hi ha els distints diagrames UML utilitzats en el disseny del projecte.

modules: Dins aquests directori hi figura els distints mòduls que confeccionen l'aplicació i que es corresponent amb les distintes capes lògiques vistes anteriorment (veure capítols posteriors).

- build.xml : fitxer on hi ha definides les tasques ant per a la construcció i desplegament de l'aplicació.
- local.properties: fitxer de configuració local on s'ha de indicar propietats globals a l'hora de definir l'aplicació. Així dins aquest fitxer s'indica el directori de desplegament de l'aplicació,

l'url de connexió de la base de dades,...

3.6 Funcionament

3.6.1 Requeriments del sistema

Els requeriments o requisits necessaris per a poder-se instalar i executar aquest projecte són els següent:

1. Connexió a Internet (recomenat):

Per a poder instal·lar i executar el simualdors estrictament no cal tenir connexió a Internet, però si s'instal·la fent un checkout del subversion si que serà necessari connexió per accedir al repositori d'Intenet.

2. Java SDK 1.6

Per tal d'assegurar-se que s'està utilitzant el compilador adient cal executar:

```
javac -version
```

i el resultat ha de ser una versió del compilador igual o superior a la 1.6. Per exemple la javac 1.6.0_16. D'ara endavant el directori on s'ha instal·lat el Java SDK l'anomenarem [$JAVA_HOME].

3. MySql

Un altre requeriment bàsic i crucial pel desplegament i funcionament del projecte és la base de dades MySql. Cal assegurar-se de tenir-la instal·lada i en funcionament. Per a instal·lar-la es pot baixar de http://www.mysql.com/downloads/ i seguir-ne les instruccions. Cal també configurar un usuari de connexió amb una clau. La versió utilitzada és la 5.1.47, però amb qualsevol de les 5.1 funcionarà.

Per a saber quina versió de MySql està instal·lat al nostre maquinari es pot provar amb aquesta comanda

```
>> mysql --version
mysql Ver 14.14 Distrib 5.1.37, for debian-linux-gnu (i486) using EditLine wrapper
```

D'ara endavant el directori on s'ha instal·lat el MySql l'anomenarem [$MYSQL_HOME].

4. Ant

Una altra eina essencial per a la instal·lació i configuració del projecte és l'ant. Aquest utensili es pot baixar directament de http://ant.apache.org/ i s'utilitza per a realitzar tasques concretes i repetitives a

l'hora de gestionar, construir i desplegar un projecte Java.

Es pot instal·lar qualsevol versió superior o igual a 1.7, encara que amb versions més velles com la 1.6 també funcionaria. Per saber si tenim instal·lat l'ant i consultar la versió ho podem fer mitjançant la següent comanda (sempre i quan es trobi al nostre path):

```
>> ant -version
Apache Ant version 1.7.1 compiled on October 19 2009
```

D'ara endavant el directori on s'ha instal·lat l'Ant l'anomenarem [$ANT_HOME].

5. Subversion (opcional)

Per a poder instal·lar i executar el simulador via *subversion*, cal fer-ho des de un ordinador amb connexió a Intenet ja que tot el codi font, llibreries necessàries i altres objectes es duu a terme accedint a un repositori d'Internet. Per a saber si tenim instal·lat correctament el *subversion* cal executar aquesta comanda:

```
>> svn --version
svn, version 1.6.5 (r38866)
compiled Aug 31 2009, 18:42:02
Copyright (C) 2000-2009 CollabNet.
Subversion is open source software, see http://subversion.tigris.org/
This product includes software developed by CollabNet (http://www.Collab.Net/).
The following repository access (RA) modules are available:
* ra_neon : Module for accessing a repository via WebDAV protocol using Neon.
- handles 'http' scheme
- handles 'https' scheme
* ra_svn : Module for accessing a repository using the svn network protocol.
- with Cyrus SASL authentication
- handles 'svn' scheme
* ra_local : Module for accessing a repository on local disk.
handles 'file' scheme
```

3.6.2 Instal·lació i configuració del projecte.

Hi ha dues maneres d'instal·lar el projecte del simulador:

1. Mitjançant els fonts entregats: Descomprimir directament el .zip en un directori de treball.
2. Realitzant un *checkout* del subversion: Un cop haver assegurat que els requisits anteriors estan correctament instal·lat a l'ordinador on es vol fer afectiva aquesta instal·lació del projecte cal fer el següent:

```
>> svn checkout http://perfilofuscator.googlecode.com/svn/branches/branch_1.0/ pfc
```

Aquest comanda fa un checkout del projecte en mode només lectura, és a dir no es permetrà pujar cap modificació al repositori.

```
>> svn checkout  http://perfilofuscator.googlecode.com/svn/trunk/ pfc
A    pfc/lib
A    pfc/lib/mysql-connector-java-5.1.12-bin.jar
A    pfc/lib/jung2-2_0_1
A    pfc/lib/jung2-2_0_1/jung-io-2.0.1.jar
A    pfc/lib/jung2-2_0_1/j3d-core-1.3.1.jar
A    pfc/lib/jung2-2_0_1/jung-api-2.0.1.jar
A    pfc/lib/jung2-2_0_1/jung-algorithms-2.0.1.jar
A    pfc/lib/jung2-2_0_1/jung-3d-demos-2.0.1.jar
A    pfc/lib/jung2-2_0_1/collections-generic-4.01.jar
... ..... .....
A    pfc/lib/jung2-2_0_1/jung-visualization-2.0.1.jar
Revisión obtenida: 2
```

Modificació de les propietats locals

A continuació cal editar el fitxer *hibernate.cfg.xml* de dins $[PFC]/etc i modificar-ne les propietats següents que fan referència a la base de dades MySql:

```
<property name="connection.url">jdbc:mysql://localhost:3306/simulador</property>
<property name="connection.username">root</property>
<property name="connection.password">root</property>
<property name="connection.pool_size">1</property>
```

3.6.3 Generació del simulador.

Un cop haver fet la descàrrega del projecte i haver modificat correctament les propietats locals, cal generar el simulador mitjançant la següent instrucció:

```
>>ant generaBuildfile: /home/joan/pfc/build.xml

prepara:

compila:
      [javac] /home/joan/pfc/build.xml:64: warning: 'includeantruntime' was not set,
defaulting to build.sysclasspath=last; set to false for repeatable builds

genera:
      [jar] Building jar: /home/joan/pfc/output/Simulador.jar

BUILD SUCCESSFUL
```

```
Total time: 16 seconds
```

Aquesta tasca ha generat un executable dins output anomenar Simulador.jar

3.6.4 Execució del simulador

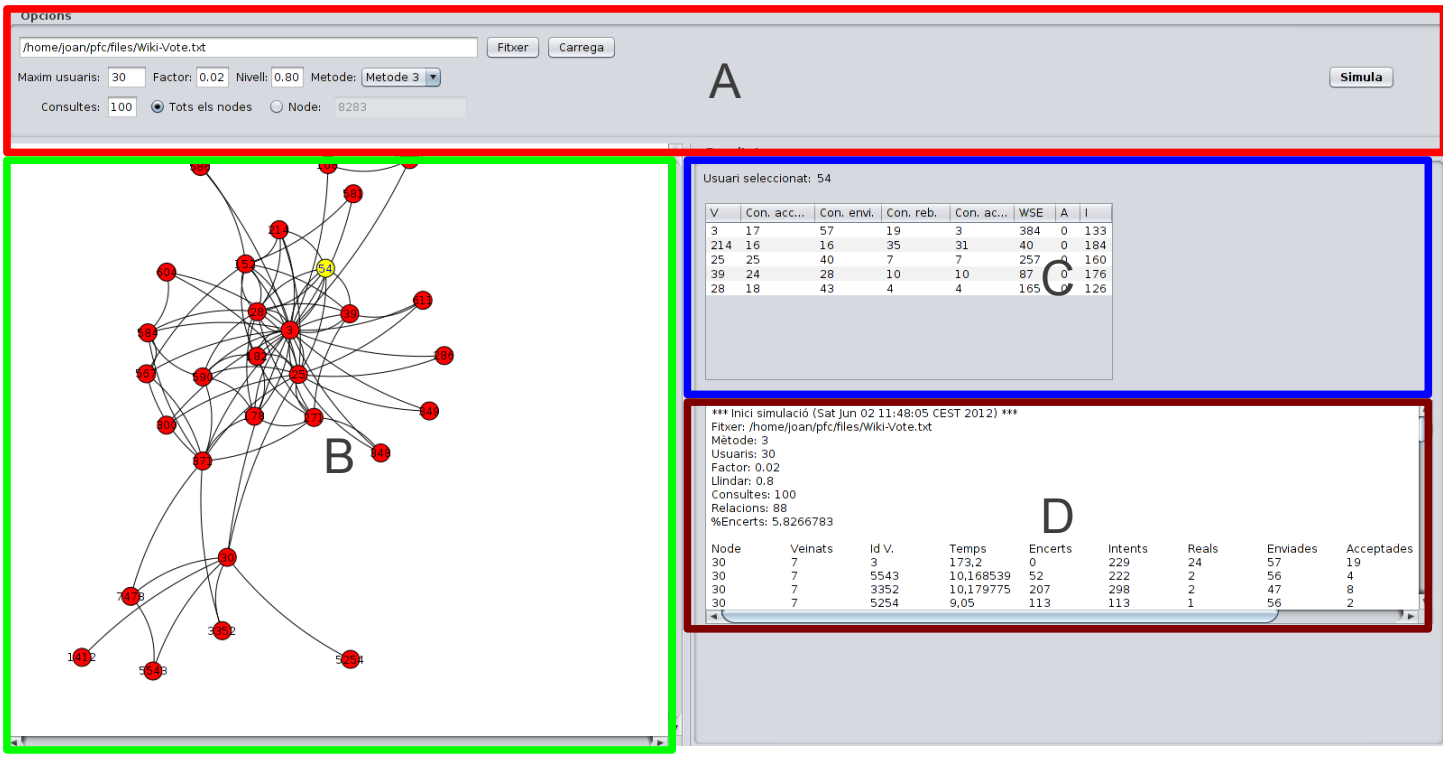
Per executar el simulador un cop hagui compilat el projecte cal executar la següent comanda:

```
>> java -jar output/Simulador.jar
```

i a continuació cap aparèixer la pantalla principal del simulador gràfic.

3.6.5 Manual de funcionament

Un cop executat la comanda anterior es llançarà la pantalla principal del simulador que és com la figura següent. Es pot veure que està dividida en quatre seccions:



A) Secció de **paràmetres** de la simulació: és on hi ha present els diferents paràmetres que es poden modificar de la simulació: fitxer de la xarxa, màxim d'usuaris, factor lambda, nivell de probabilitat, mètode d'estimació, consultes generades, nodes (o bé tots els que conformen la xarxa) qui genera les consultes.

B) Secció de **visualització de la xarxa**: un cop carregada la xarxa es visualitza en aquesta

secció. Aquest panell és interactiu de manera que, un cop simulat el comportament, en situam damunt un node (fent click a sobre d'ell) podem visualitzar els resultats concrets d'aquest node.

C) Secció de **resultats del node**: en aquesta secció hi apareixen els resultats d'un node determinat, seleccionat en la secció B. Es mostren les següents dades: Veïnat, consultes acceptades del veïnat, consultes rebudes del veïnat, consultes acceptades per part del veïnat, consultes enviades al veïnat, consultes que el veïnat envia al WSE, encerts i intents.

D) Secció de **resultats globals de la xarxa**: En aquest parell hi apareixen els resultats globals de la simulació

4 Simulacions

4.1 Introducció

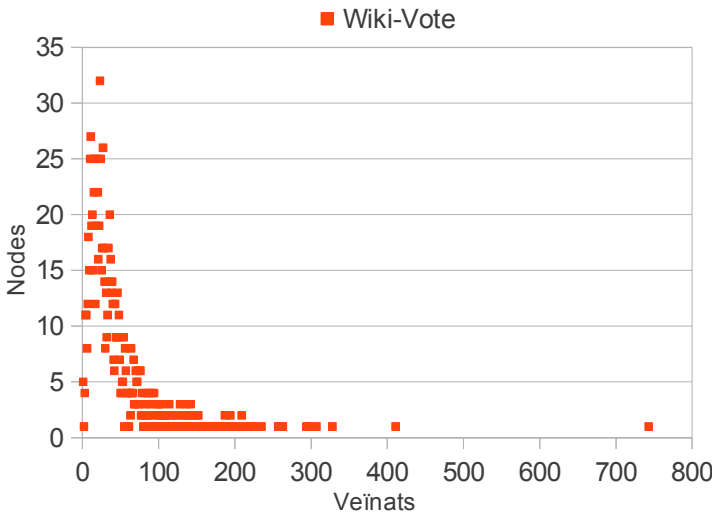
El propòsit de les simulacions és per una banda primer estudiar el comportament dels distints mètodes d'estimació de la xarxa i per l'altre l'estudi del comportament del nostre sistema, utilitzant una nova variable: el temps mitjà en que rebem consultes dels veïnats.

4.2 Xarxes estudiades

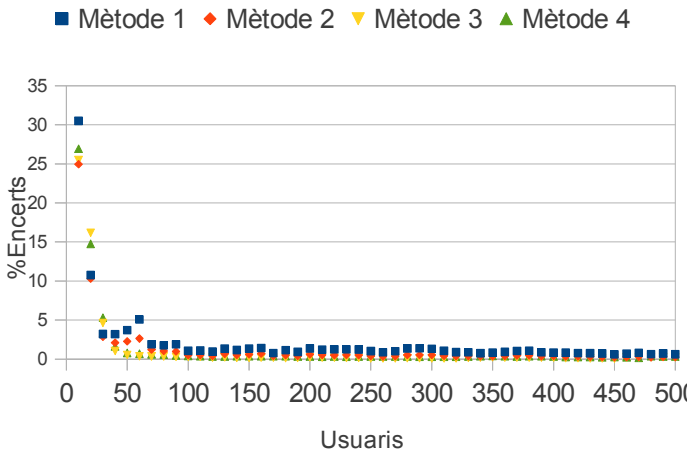
En les simulacions estudiades s'han analitzat quatre xarxes distintes totes elles escollides de *Stanford Large Network Dataset Collection* [1]. A continuació descriurem breument cada un dels conjunts de dades de les xarxes escollides.

4.2.1 Xarxa 1: Wiki Vote

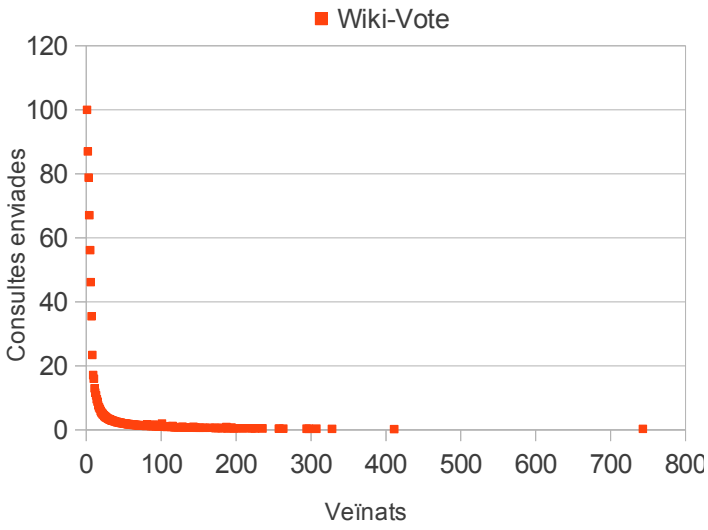
Wiki Vote que conté totes les dades de la votació de Wikipedia des de l'inici de la *Wikipedia* fins a gener de 2008. A la gràfica següent podem veure com es van distribuint els 500 primers nodes. En ella es pot veure com el nombre de nodes amb menys de 50 veïnats és major que la resta de la xarxa, i que a mesura que augmentam el nombre de veïnats, disminueix la seva aparició.



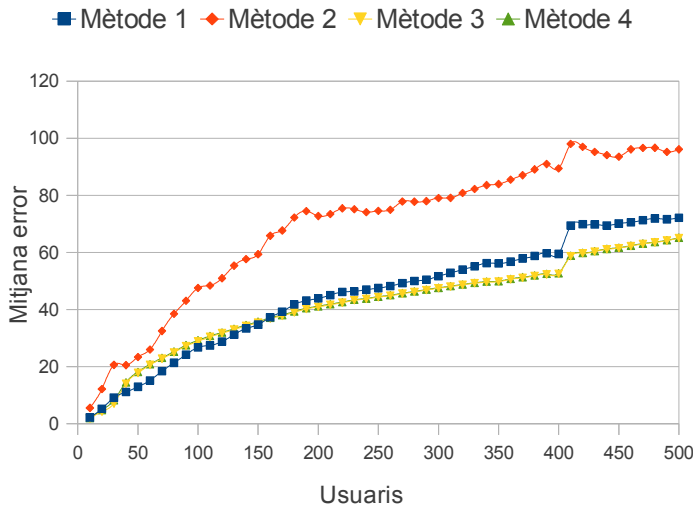
Il·lustració 1: Densitat Wiki-Vote



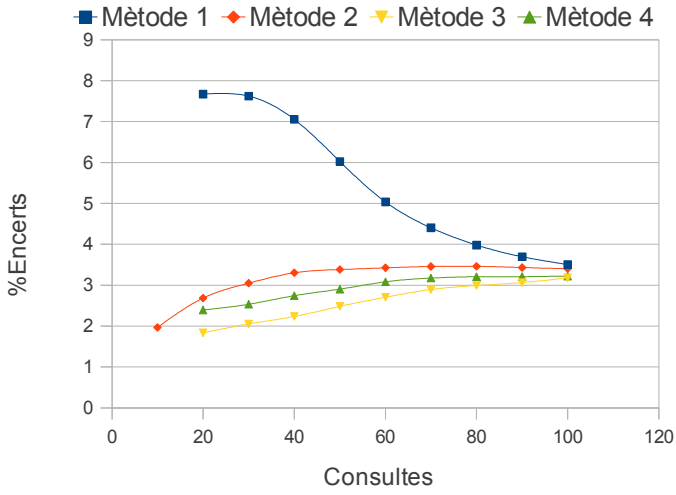
Il·lustració 2: %Encerts mètodes a Wiki-Vote



Il·lustració 3: Consultes rebudes/veïnat



Il·lustració 4: Mitjana error



Il·lustració 5: %Encerts/Consultes

Com es pot veure a la primera gràfica es pot aproximar a una una gràfica *power-law*, característica de les xarxes *scale-free.* Una xarxa *scale-free* és una xarxa social on el grau de distribució dels seus nodes segueix una funció power-law, com a mínim asimptòtica. Això és la fracció P(k) des nodes en la xarxa amb k connexions a altres nodes va segons la següent funció:

$$P(k) \sim ck^{-\gamma}$$

on c és una constant normalitzada i l'exponent lambda està típicament entre 2 i 3.

A la següent gràfica es mostra el percentatge d'encerts a l'hora de fer l'estimació del nombre de veïnats per cada un dels mètodes implementats en el simulador. Les dades obtingudes provenen de simulacions fetes amb un màxim de 500 nodes, amb un factor *lambda* de 0,02. Es pot veure que per a una quantitat de nodes menor que 50, els mètode 3 i 4 són els que obtenen millor rendiment, mentre que a mesura que augmenta el nombre de nodes el rendiment baixa i es va igualant entre els distints mètodes.

A la il·lustració 3 es pot veure el comportament del sistema comparant el nombre mitjà de consultes rebudes del veïnat amb el nombre de veïnats. Es pot veure com per a nombres petits de veïnats el nombre de consultes rebudes és elevat, mentre que va disminuint a mesura que es té més veïnats. Això és una conseqüència directa del protocol, que intenta distribuir totes les consultes amb els seus nodes veïnats, i en el cas de tenir pocs veïnats, aquests reben moltes consultes.

A la il·lustració 4 hi ha representat la mitjana d'error (**veïnats reals – veïnats estimats**). Es pot comprovar que els que menor nivell tenen són les mètode 3 i 4 i el mètode amb més nivell d'error és el mètode 1.

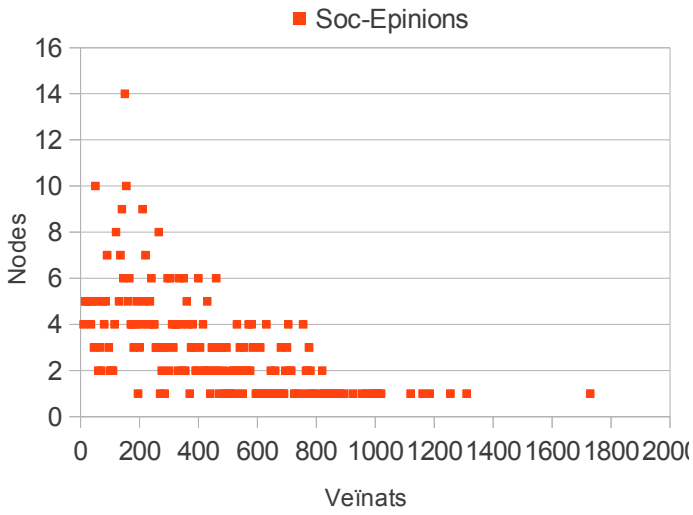
A la il·lustració 5 hi ha representat el % d'encerts segons el nombre de consultes generades a la simulació. Es pot comprovar com efectivament a mesura que generam més consultes, els mètode 2,3 i 4 responen més bé, mentre que el mètode 1 decau.

Taula 1: %Encerts Wiki-Vote

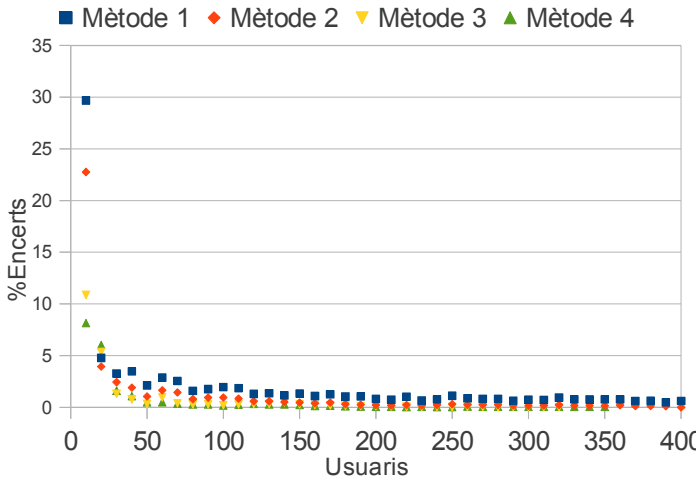
Mètodes	**Entre 10 i 20 nodes**		**Entre 20 i 50 nodes**		**Entre 50 i 100 nodes**		**Entre 100 i 200 nodes**		**Entre 200 i 500 nodes**	
	%	Desviació	%	Desviació	%	Desviació	%	Desviació	%	Desviació
Mètode 1	15,8	4,4	3,88	11,4	2,22	21,22	1,15	36,6	0,89	61,31
Mètode 2	14,03	10,71	2,74	18,95	1,24	29,54	0,53	45	0,36	66,85
Mètode 3	18,54	3,88	2,06	15,18	0,34	25,5	0,20	36	0,14	52,02
Mètode 4	17,86	4,23	2,38	15,3	0,55	25,5	0,36	36	0,31	53,18

4.2.2 Xarxa 2: Soc-Ephinions

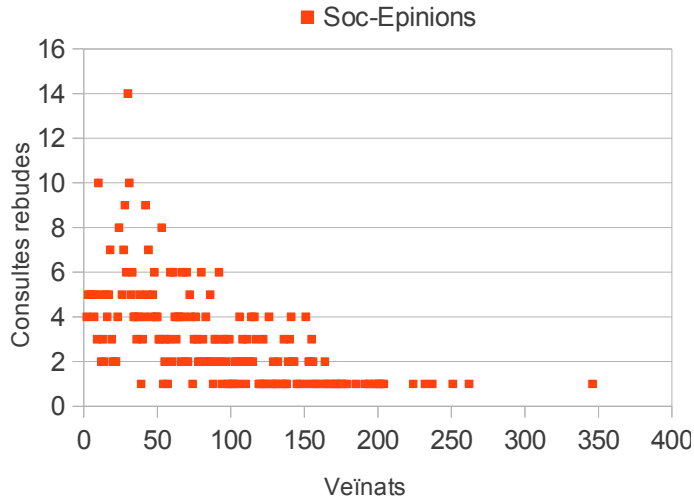
Aquesta xarxa està confeccionada a partir d'un conjunt de dades extret d'una xarxa social d'un lloc general on els consumidors de *Epinions.com* aporten la seva opinió. Els membres del lloc poden decidir donar la confiança a un o a altre consumidor. Totes les relacions de confiança interactuar i formar la xarxa de confiança que després es combinen amb notes de revisió per a veure els comentaris que mostren a l'usuari. A la gràfica següent podem veure com es van distribuint els 500 primers nodes Es pot apreciar com també segueix una estructura *power-law* on les majors freqüències d'aparició ocorren amb nombre de veïnats petits.



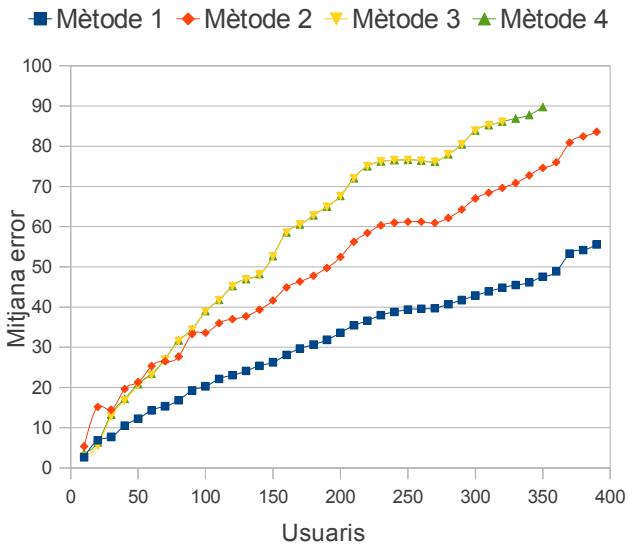
Il·lustració 6: Densitat



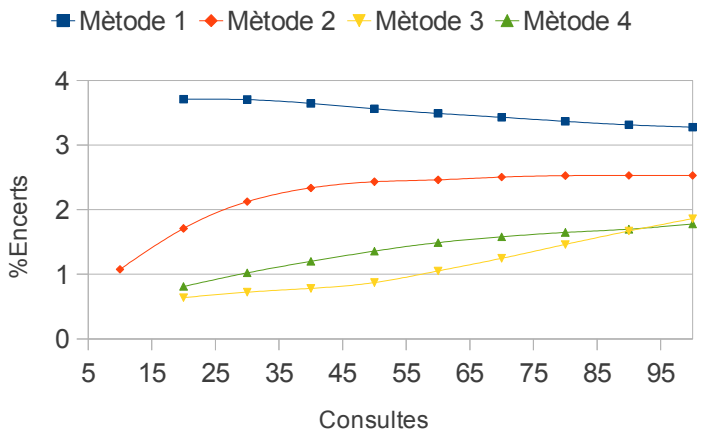
Il·lustració 7: %Encerts



Il·lustració 8: Consultes/veïnats



Il·lustració 9: Mitjana error



Il·lustració 10: %Encerts/consultes

A diferència de les altres xarxes, aquesta és la que presenta un nivell més alt de relacions, d'aquí que el temps per simular-la sempre ha estat el més gran. Es pot veure com l'estructura power-law no està tan ben definida com en el cas anterior.

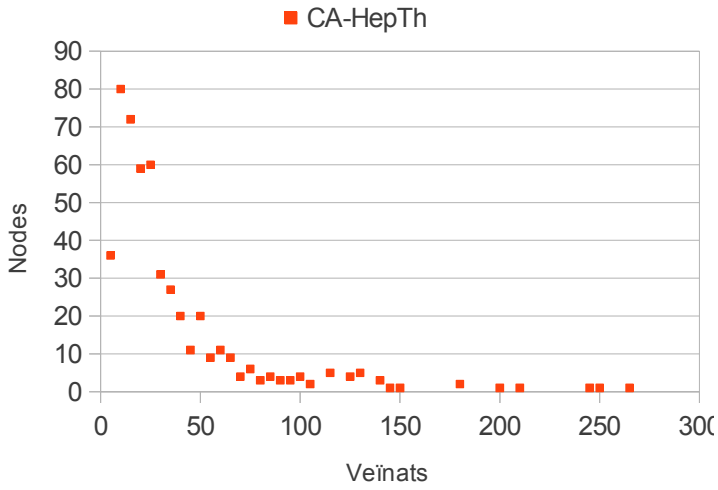
Per aquesta xarxa, en general, podem dir que el mètode d'estimació que més alt nivell d'encerts ens dóna i més baix nivell d'errors és el mètode 1.

Taula 2: %Encerts Soc-Epinions

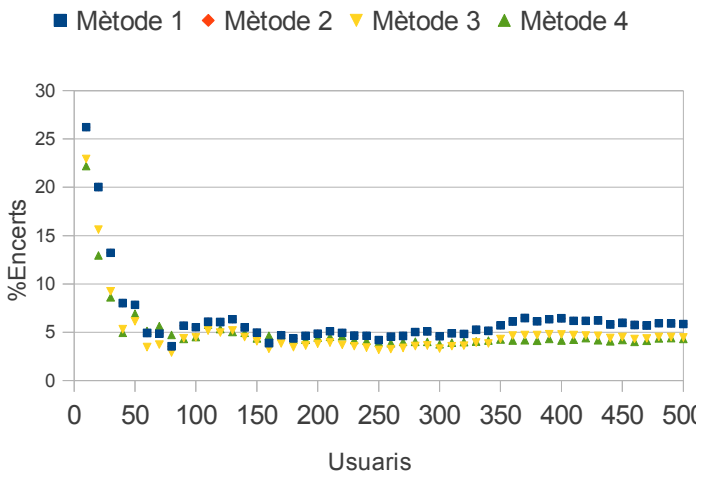
Mètodes	**Entre 10 i 20 nodes**		**Entre 20 i 50 nodes**		**Entre 50 i 100 nodes**		**Entre 100 i 200 nodes**		**Entre 200 i 500 nodes**	
	%	Desviació	%	Desviació	%	Desviació	%	Desviació	%	Desviació
Mètode 1	12,18	5,5	2,89	10,5	2	17,47	1,19	28,39	0,75	45
Mètode 2	9,5	11,9	1,73	18,2	1	28,91	0,43	49,72	0,18	77
Mètode 3	7,0	5,19	0,92	17,69	0,34	32,09	0,21	51,6	0,04	78
Mètode 4	6,6	5,7	1,19	17,85	0,30	32,16	0,16	51,7	0,05	81

4.2.3 Xarxa 3: CA-HepTh

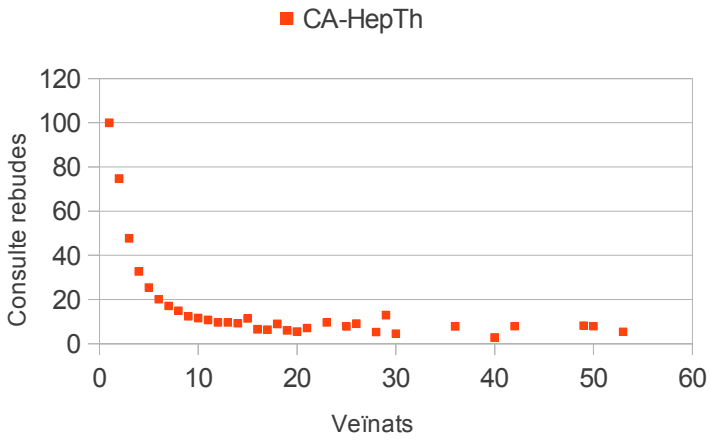
Arxiu HEP-TH (Teoria física d'altes energies) és xarxa de col·laboració de la arXiv e-print i abasta col·laboracions científiques entre els documents presentats als autors Física d'Altes Energies - Teoria de la categoria



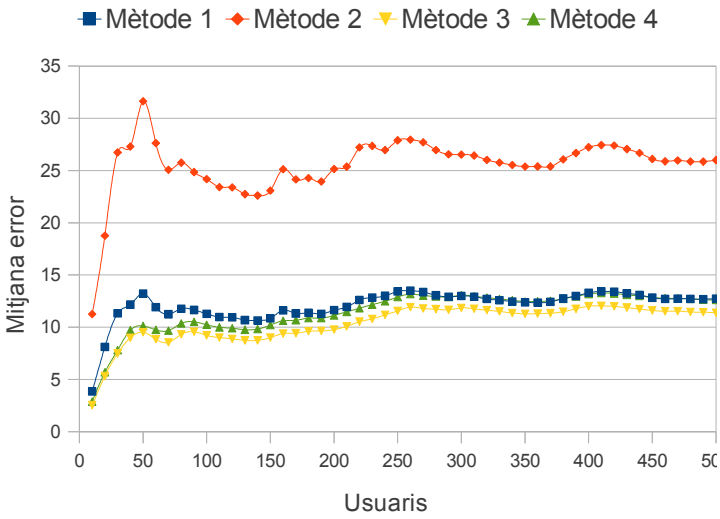
Il·lustració 11: Densitat



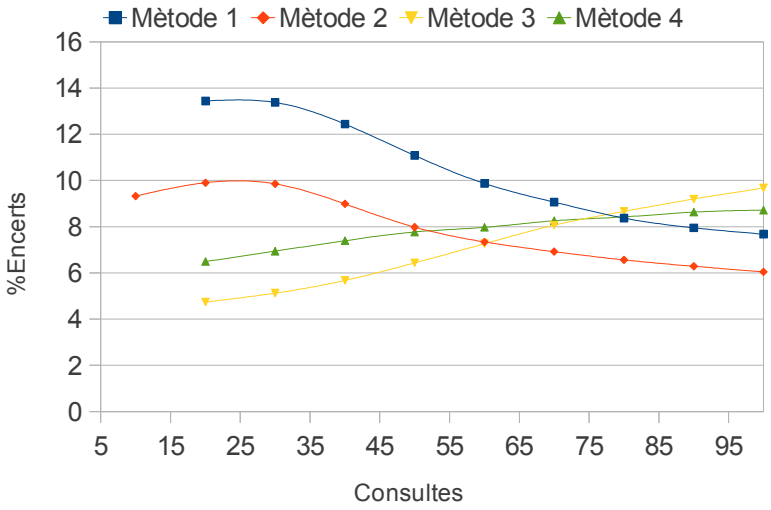
Il·lustració 12: %Encerts



Il·lustració 13: Consultes/veïnats



Il·lustració 14: Mitjana error



Il·lustració 15: %Encerts/consultes

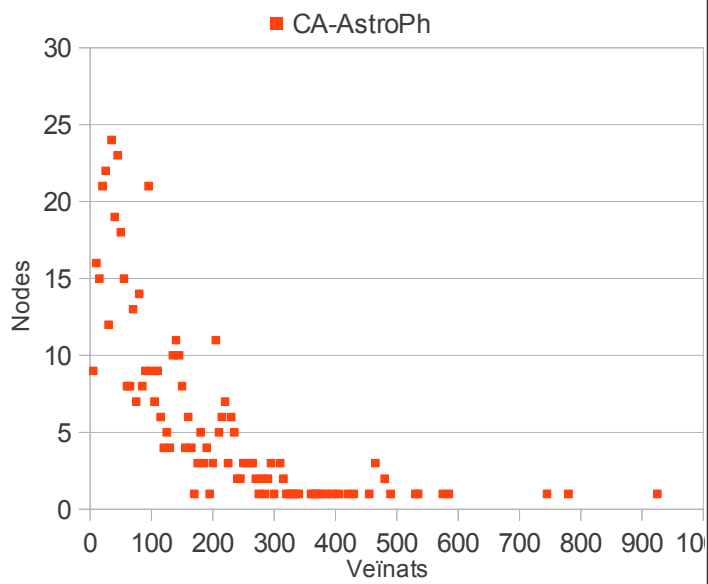
Aquesta xarxa és la de menor nombre d'usuaris i de relacions que hem simulat. Es pot comprovar com el perfil power-law és molt marcat. Els percentatges d'encerts son molt semblant entre els tres mètodes, encara que reacciona millor el mètode 1 (s'entén donat que té poca quantitat de nodes). Si es mira la gràfica de la mitjana d'error, el mètode que pitjor respon amb diversos usuaris és el mètode 2.

Taula 3: %Encerts CA-Hep.Th

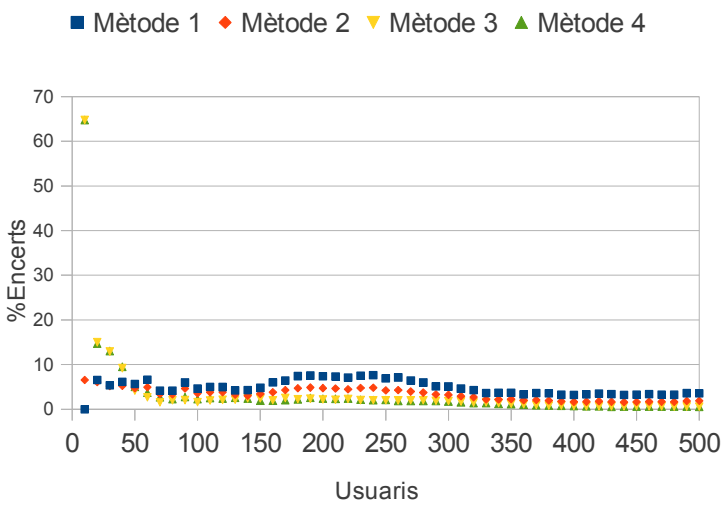
Mètodes	**Entre 10 i 20 nodes**		**Entre 20 i 50 nodes**		**Entre 50 i 100 nodes**		**Entre 100 i 200 nodes**		**Entre 100 i 500 nodes**	
	%	Desviació	%	Desviació	%	Desviació	%	Desviació	%	Desviació
Mètode 1	21,66	6,9	10,10	12,00	5,24	11,71	5,03	11,19	5,5	12,85
Mètode 2	17,56	16,15	7,37	26,39	4,08	23,4	4,089	21,01	4,18	23,19
Mètode 3	18,73	4,61	6,16	8,86	4,05	9,77	4,9	9,52	4,41	11,52
Mètode 4	15,4	4,93	7,16	9,26	4,9	10,3	4,6	10,4	4,15	12,75

4.2.4 Xarxa 4:CA-AstroPh

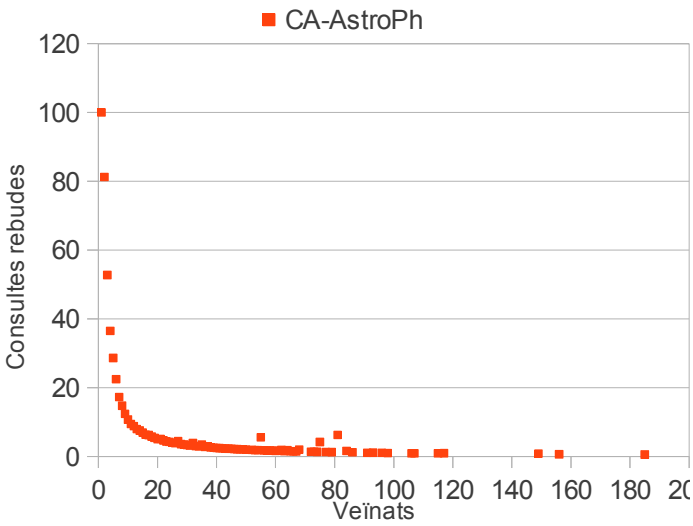
Xarxa 4: Arxiu *astre-ph* (astrofísica) xarxa de col·laboració és de la arXiv e-print i cobreix les col·laboracions científiques entre els documents presentats als autors de la categoria de Física de 'Astres



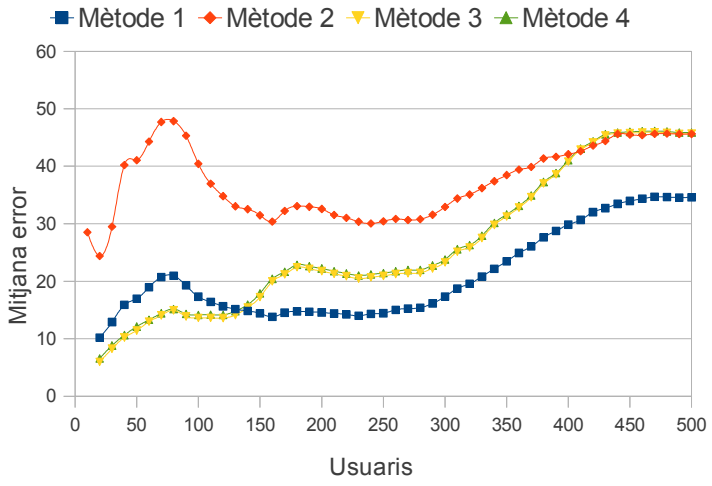
Il·lustració 17: Densitat



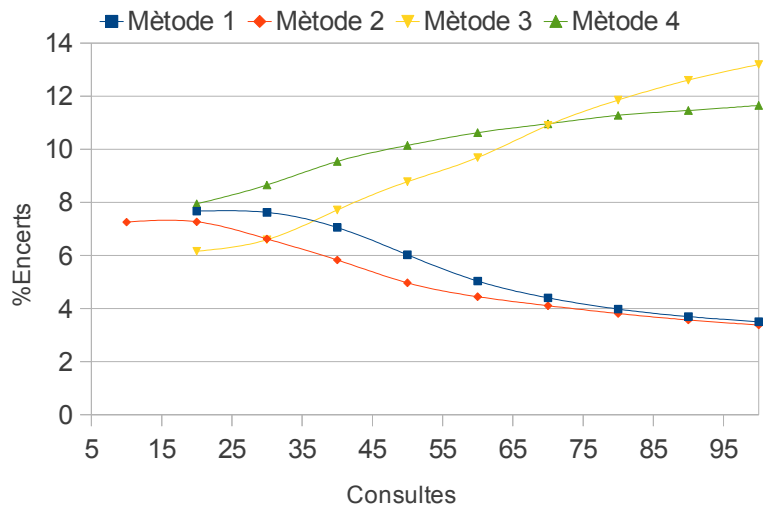
Il·lustració 18: %Encerts/usuaris



Il·lustració 19: Consultes/veïnats



Il·lustració 16: Mitjana error



Il·lustració 20: %Encerts/consultes

Aquesta xarxa és molt òptima per a compara els mètodes d'estimació. Segueix una estructura de densitat power-law i els mètodes que sobresurten pel seu % d'encerts són el mètode 3 i el 4. En aquesta xarxa s'observa que quan augmentam el nombre de consultes, aquests dos mètodes tendeixen a reaccionar millor, mentre que els mètode 1 i2 decreixen.

Taula 4: %Encerts CA-Astro.Ph

Mètodes	**Entre 10 i 20 nodes**		**Entre 20 i 50 nodes**		**Entre 50 i 100 nodes**		**Entre 100 i 200 nodes**		**Entre 200 i 500 nodes**	
	%	Desviació	%	Desviació	%	Desviació	%	Desviació	%	Desviació
Mètode 1	5,1	9,7	5,81	15	5,08	19,5	6,06	14,8	3,94	28.03
Mètode 2	6,1	27,5	5,05	35	3,9	31,81	4,08	26,9	2,21	38
Mètode 3	25,66	5,6	8,2	10,24	2,11	14,10	2,02	17,44	1,008	37
Mètode 4	25,42	5,9	8,6	10,64	2,84	14,40	2,25	17,88	0,9	38

4.2.5 Resum resultats

En resum, les quatre xarxes simulades presenten una estructura tipus *power-law*, on per a un nombre de nodes relativament baix (<50 nodes) s'obté que el mètode 3 proporciona el % d'encerts més elevat. Quan augmentam el nombre de nodes de la xarxa el rendiment disminueix i l'eficàcia dels mètodes s'iguala.

A continuació introduirem una nova variable a tenir en compte per a intentar millorar aquest rendiment a l'hora de fer una predicció del nombre de veïnats. Aquesta variable és el temps mitjà en rebre consultes dels nodes veïnats.

Taula 5: Xarxes simulades

Xarxa	**Nodes**	**Relacions**	**Coeficient agrupació[2]**	**Diàmetre[3]**	**Diàmetre efectiu 90-percentil**
Wiki-Vote	7.115	103.689	0,2089	7	3,8
Soc-Epinions	75.879	508.837	0,2283	13	5
CA-Hep.Th	9.877	51.971	0,4714	17	7,5
CA-AstroPh	18.772	396.160	0,6306	14	5,1

Taula 6: %Encerts globals

Mètodes	**Entre 10 i 20 nodes**		**Entre 20 i 50 nodes**		**Entre 50 i 100 nodes**		**Entre 100 i 200 nodes**		**Entre 200 i 500 nodes**	
	%	Desviació	%	Desviació	%	Desviació	%	Desviació	%	Desviació
Mètode 1	11,6	6,4	3,77	11,7	2,39	18,22	1,8	27,31	1,71	43,17
Mètode 2	10,1	15,38	3,05	23,28	2,00	33,7	1,4	46,29	1,22	64
Mètode 3	20,05	4,6	4,45	13,9	1,51	24,1	1,02	41,97	0,88	51,9
Mètode 4	16,59	5,16	4,19	14,41	1,52	24,9	0,93	42,24	0,78	55,21

2 Coeficient d'agrupació (clustering coefficient) de un node a una xarxa social quantifica el grau d'agrupament (interconnexió) amb els seus veïnats.

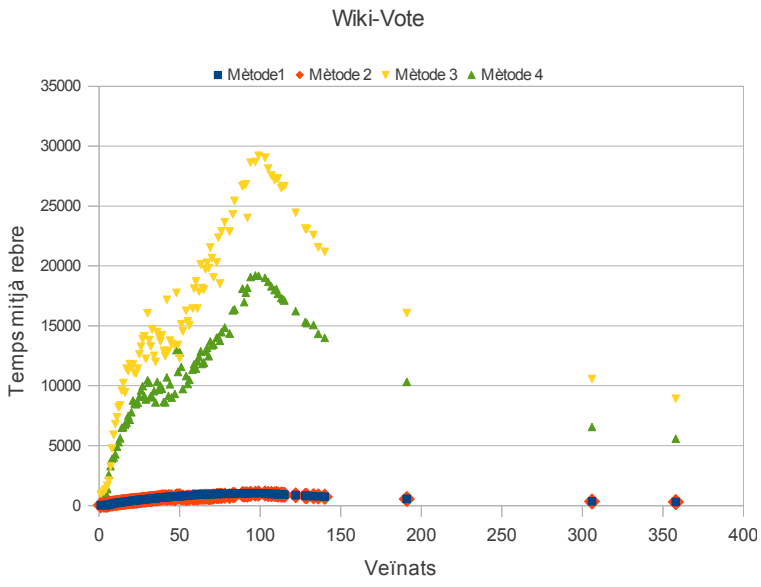
3 El diàmetre d'una xarxa social és la longitud (relacions) de la trajectòria més llarga entre dos nodes qualssevol.

4.3 Temps mitjà en rebre consultes d'un veïnat

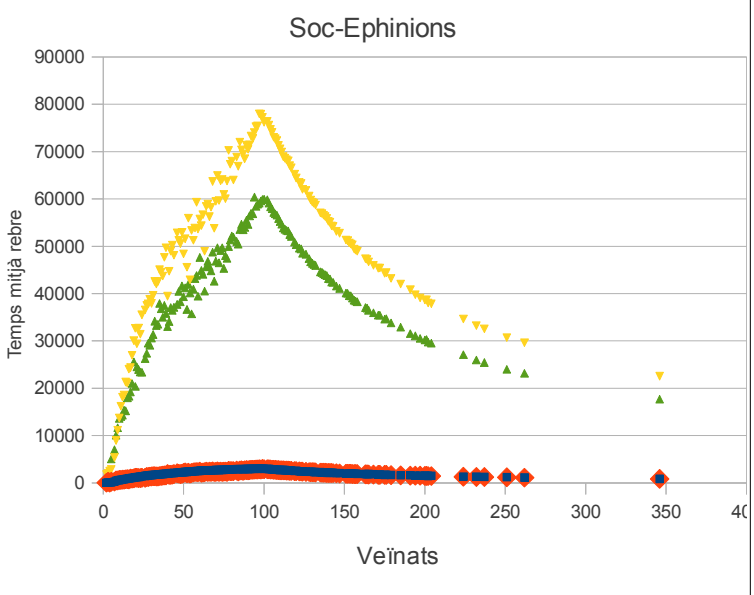
Al llarg del projecte s'ha vist que els mètodes d'estimació del nombre de veïnats d'un determinat node fan servir bàsicament les següents variables per a predir-ho: nombre propi de veïnats que el node presenta, nombre mitjà de consultes rebudes del veïnat. S'ha vist, arran de les diferents simulacions fetes, que aquests mètodes són poc eficients (presenten un baix rendiment) amb nombre de nodes alt, o mitjanament alt (>100, per exemple). En l'intent de millorar-los es vol fer l'estudi d'una nova variable que permetrà estudiar aquest problema: el temps mitjà en que rebem les consultes dels veïnats.

Intuïtivament podem pensar que com més alt sigui aquest temps mitjà a rebre consultes del veïnat és per què aquest veïnat presenta un nombre alt de relacions amb altres nodes. I podem pensar també que com menor sigui aquest temps mitjà menor serà el nombre de connexions que presenti.

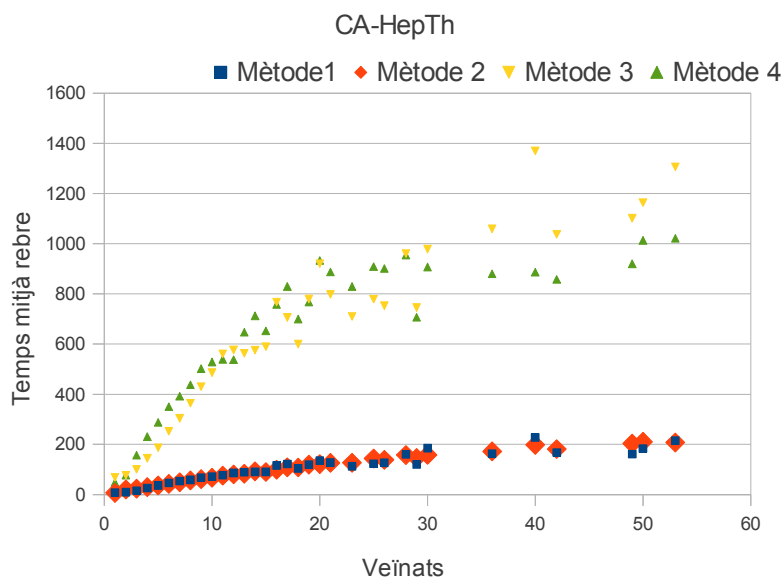
Com s'ha comentat anteriorment hem introduït una nova variable per tal d'intentar millorar l'eficiència dels mètodes utilitzats a l'hora d'estimar el nombre de veïnats d'un determinat node. Aquesta variable és el **temps mitjà** en que un determinat node rep consultes del seu veïnat. Fent una simulació amb les quatre xarxes presentades anteriorment s'obté la següent gràfica, on s'ha representat el temps mitjà en rebre consultes segons el nombre de veïnats del node. Es pot apreciar que a mesura que augmentam el nombre de veïnats, el temps mitjà per a rebre consultes també augmenta. Totes aquestes dades estan calculades utilitzant el mètode 3 que és el que ens donava millors rendiments per a pocs nodes.



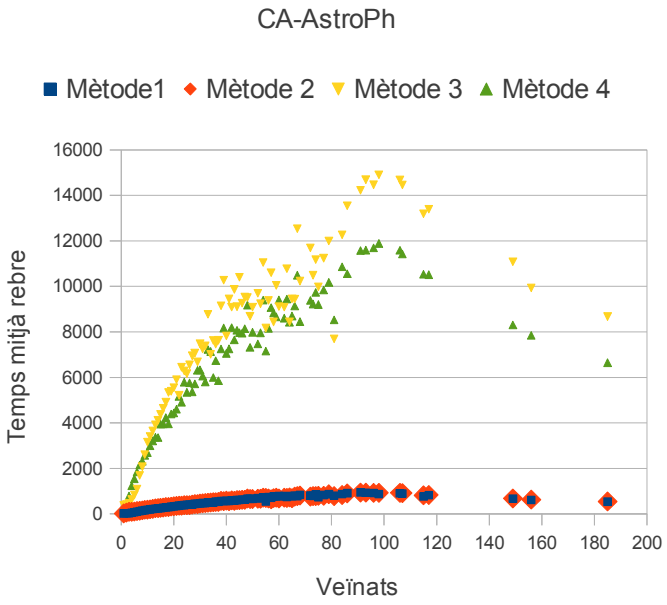
Il·lustració 21: Temps mitjà/veïnats (Wiki-Vote)



Il·lustració 22: Temps mitjà/veïnats (Soc-Epinions)



Il·lustració 23: Temps mitjà/veïnats (CA-HepTh)



Il·lustració 24: Temps mitjà/veïnats (CA-AstroPh)

Cal tenir en compte, que molts cops, per un nombre alt de veïnats (per exemple 360) ens dóna un temps mitjà relativament petit i és degut a que el nombre de notes amb aquest nombre alt de veïnats és molt baix (propietat de les xarxes power-law) i per tant el temps mitjà també és petit.

Com s'observa a la gràfica la xarxa *CA-HepTh* és la que presenta un temps mitjà d'accés més constant i baix. Això és degut a la poca quantitat de nodes presents i a també a la quantitat relativament baixa de relacions entre ells.

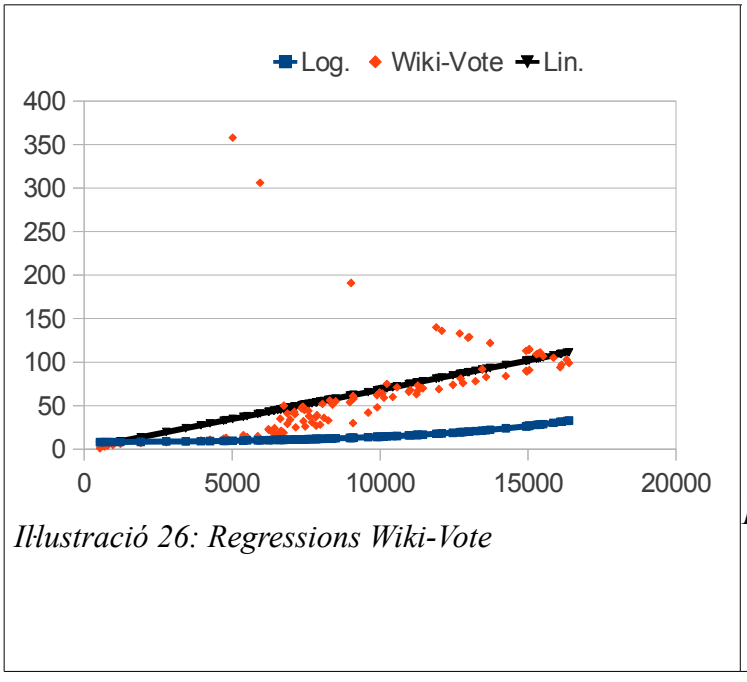
A les altres tres xarxes s'hi pot veure un comportament on fins als 100 veïnats augmenta ràpidament el temps mitjà en rebre consultes. Després aquest temps s'estabilitza. Aquest fet de disminuir, com ja s'ha comentat és degut a que el nombre de nodes amb aquest nombre alt de veïnats és molt baix (propietat de les xarxes *power-law*) i per tant el temps mitjà també és petit.

4.4 Aproximacions al nombre de veïnats a partir del temps mitjà

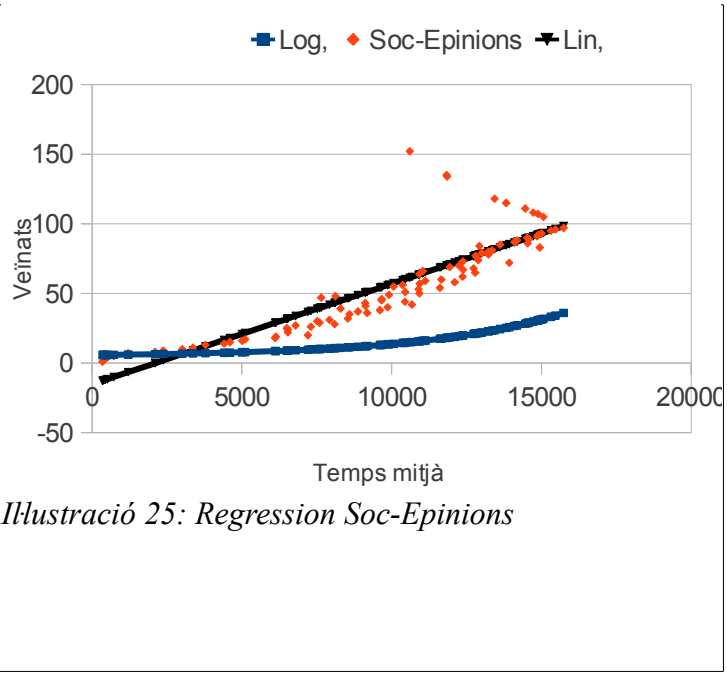
A partir de la idea anterior s'ha calculat la gràfica del nombre de veïnats en funció del temps mitjà en rebre consultes per a cada simulació. A continuació per a cada simulació es calcula la recta de regressió lineal i logarítmica. Cal tenir en compta que aquestes regressions són vàlides per aquella simulació en concret i per tant depèn:

- Usuaris màxims de la xarxa.
- Consultes generades.
- Factor lambda escollit
- Nivell de probabilitat escollit
- Mètode d'estimació escollit: pel nostre cas sempre es calcula a partir de dades obtingudes amb el mètode 3, que és el que es vol millorar.

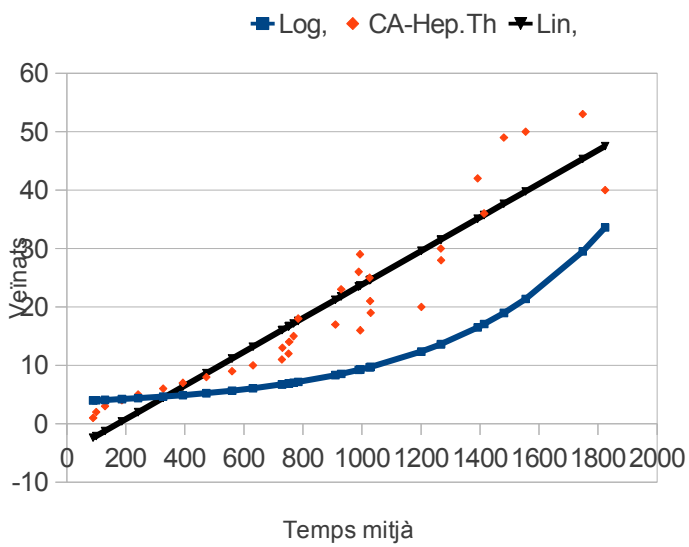
A continuació s'ha calculat les regressions lineals i logarítmiques per a les quatre xarxes comentades anteriorment. Aquest procediment és solament un exemple, ja com que com hem dit abans



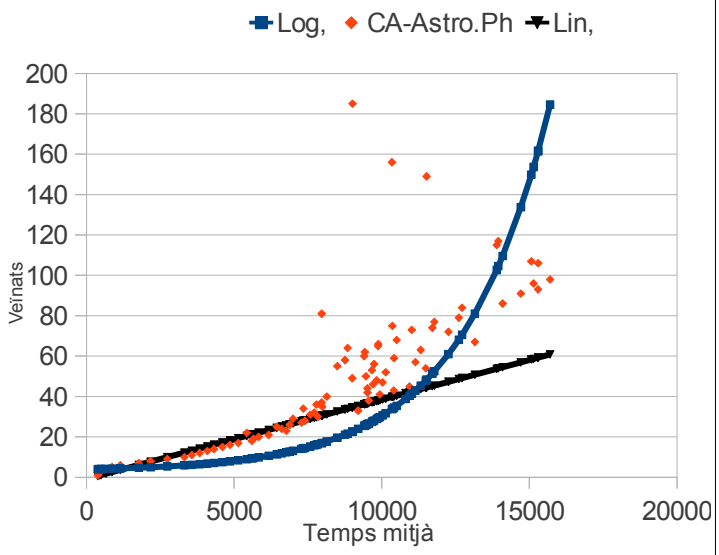
Il·lustració 26: Regressions Wiki-Vote



Il·lustració 25: Regression Soc-Epinions



Il·lustració 27: Regressions CA-Hep.Th



Il·lustració 28: Regressions CA-Astro.Ph

Per cada simulació s'ha de calcular prèviament el seu model de regressió.

Es pot apreciar com la pendent de la recta de regressió (tan lineal com logarítmica) és creixent en les quatre xarxes. A partir d'aquestes estimacions es vol crear un nou mètode d'estimació del nombre de veïnats d'un determinar node de la xarxa social. Anomenarem aquest mètode: **mètode 5**.

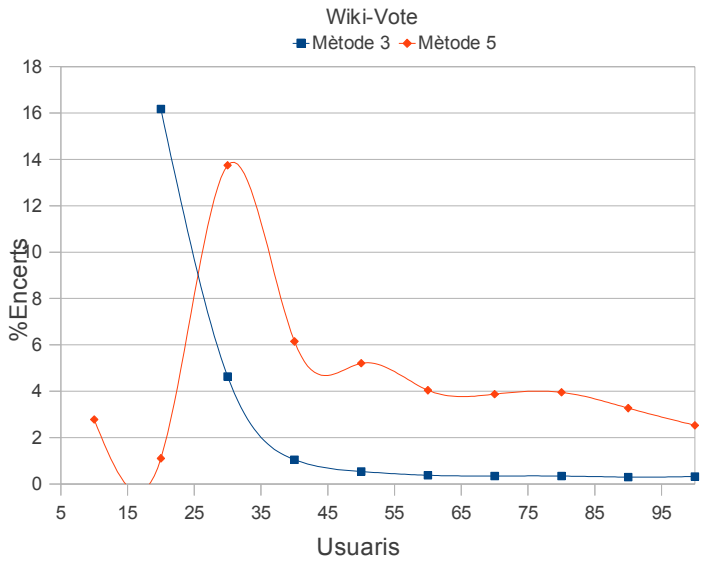
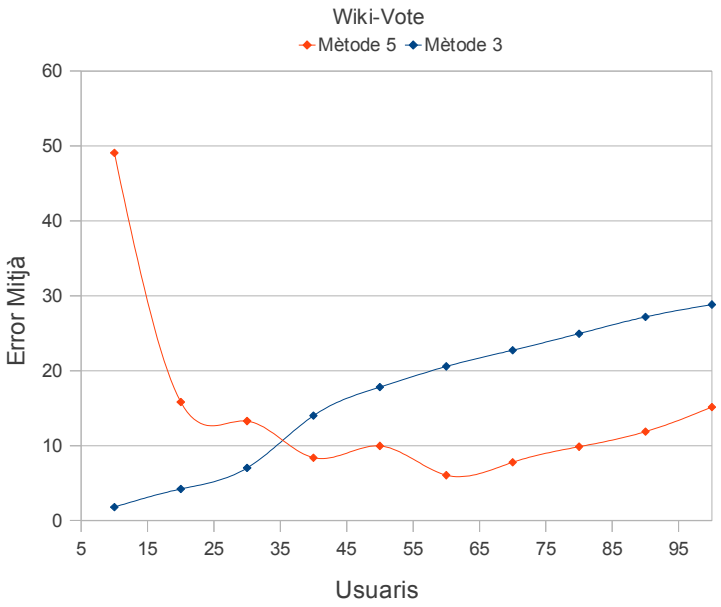
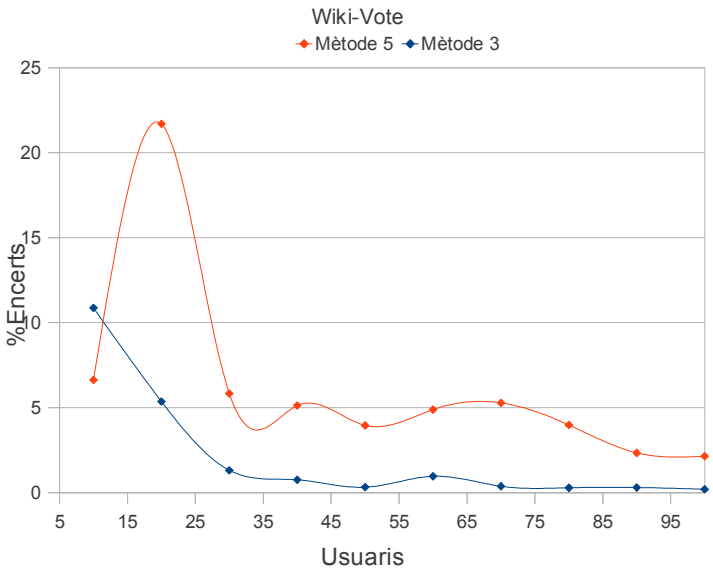
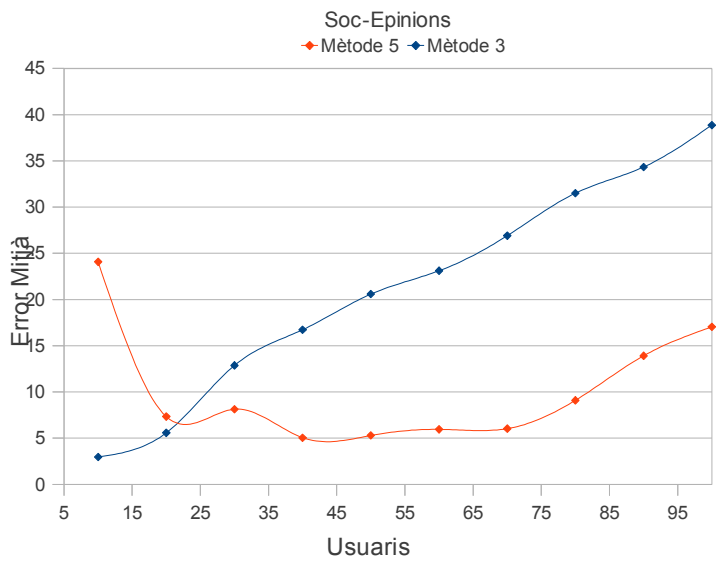
El sistema haurà de tenir en compte els dos punts següents:

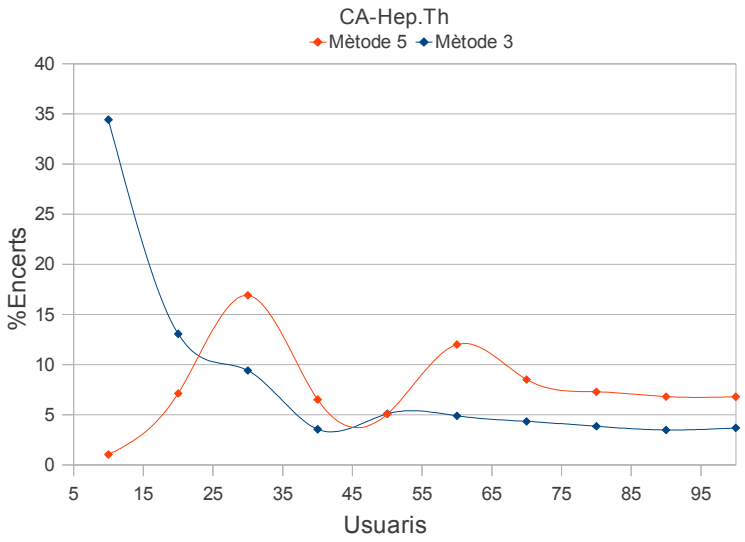
- Utilitzarem la **regressió lineal** i no la logarítmica ja que s'ajusta més a les dades i la complexitat de càlcul és menor.
- El sistema haurà de **calcular la regressió abans de començar a simular.** Per tant necessitam que hi hagi les suficients dades simulades dins el nostre magatzem.
- Calcula les regressions segons el temps mitjà en rebre consultes basats en el **mètode 3**.
- Si per alguna raó no es pot fer l'estimació amb aquest mètode, utilitza el mètode 3.
- No s'ha de calcular la regressió per a cada intent d'estimació del node, sinó que partirem d'una recta general.

Mitjançant aquestes premisses ens asseguram que la complexitat computacional del mètode sigui la mateixa que la del mètode 3, és a dir **O(lon n)**. A l'apartat següent podem veure el comportament d'aquest nou mètode mitjanament dues gràfiques per cada xarxa: una amb el %Encerts i l'altre la mitjana d'error.

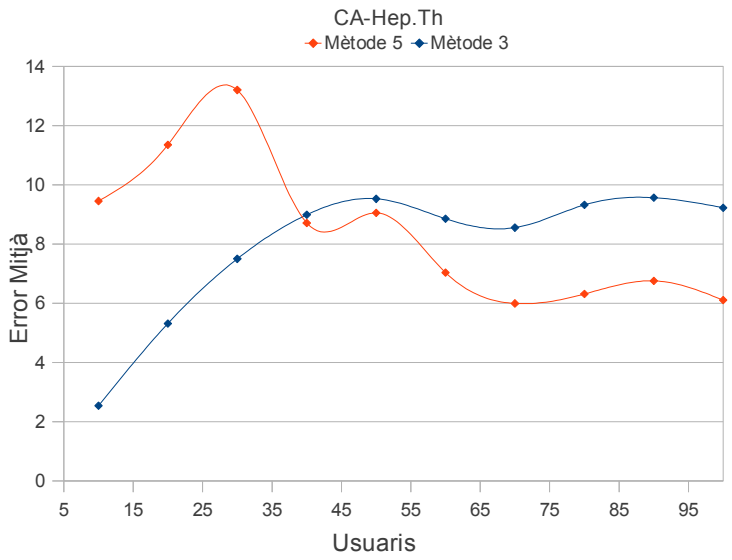
4.5 Resultats simulacions temps mitjà en rebre consultes.

A les quatre gràfiques següents es pot veure els resultats obtinguts a partir de simulacions amb les quatre xarxes seleccionades. Es pot veure representat el % d'encerts en funció del nombre de nodes que intervenen a la xarxa. Hi ha representats els dos mètodes: mètode 3 i el mètode 5 (nou mètode a partir de les rectes de regressió i el temps mitjà en rebre consultes)

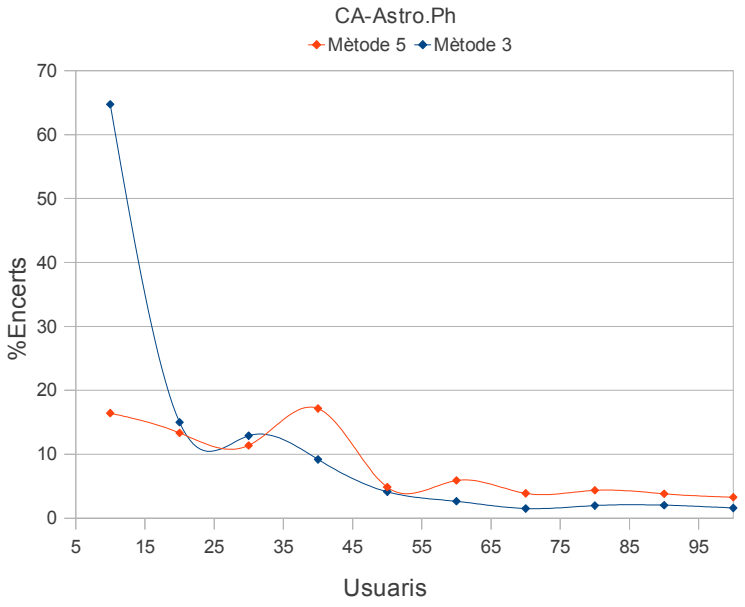
%Encerts	Error mitjà
	
Il·lustració 29: %Encerts mètode 5 (Wiki-Vote)	*Il·lustració 30: Mitjana error mètode 5 (Wiki-Vote)*
	
Il·lustració 31: %Encerts mètode 5 (Soc-Epinions)	*Il·lustració 32: Mitjana error mètode 5 (Soc-Epinions)*



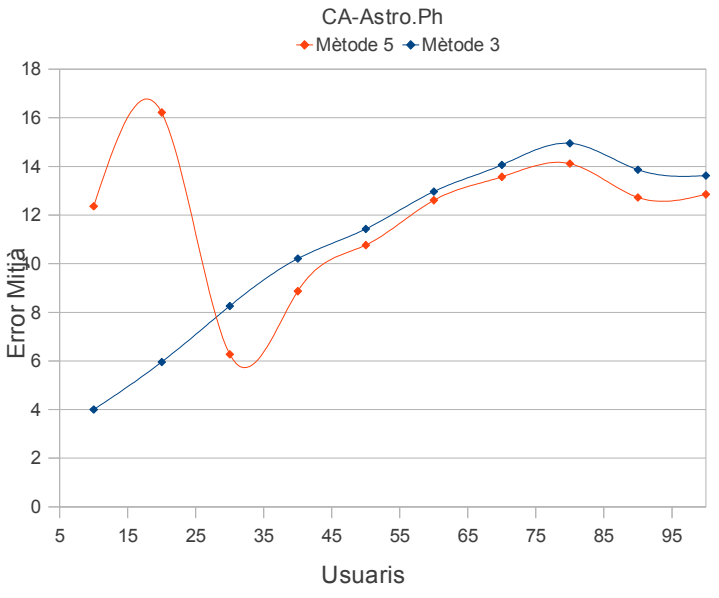
Il·lustració 33: %Encerts mètode 5 (CA-He.Th)



Il·lustració 34: Mitjana error mètode 5 (CA-Hep.th)



Il·lustració 35: %Encerts mètode 5 (CA-Astro.Ph)



Il·lustració 36: Mitjana error mètode 5 (CA-Astro.Ph)

A partir del conjunt de gràfiques anterior podem extreure les següents conclusions sobre aquest mètode d'estimació de la mida la xaxa:

- Globalment es pot dir que **millora el rendiment** del mètode 3: En les quatre gràfiques l'error mitjà del mètode 5 (per gairebé tots els usuaris) està per sota del del mètode 3. Quan al % d'eficàcia també està per sobre del mètode 3.

- Per a un màxim d'usuaris inferior 20 el mètode 3 continua aproximant més bé, però quan augmentam aquesta xifra el % d'encerts amb el mètode 5 és bastant més bo. Cal dir que en totes les simulacions fetes (variant gairebé tots els paràmetres) no he aconsegüit rendiments per sota de 1%, mentre que amb el mètode 3, per exemple a Wiki Vote, amb només 50 usuaris ja baixam d'aquest llindar de l'1%.
- L'error mitjà també té un comportament semblant per nombre d'usuaris màxim < 20. El mètode 3 presenta unes xifres més baixes que el mètode 5. Però de igual manera si augmentam aquest nombre d'usuaris, la mitjana d'error també es situa ràpidament per sobre de les dades del mètode 5.

Per contrapartida cal remarcar el següent inconvenients d'aquest mètode:

- Per calcular la regressió que ens aproximarà el model a una recta en funció del temps mitjà en rebre les consultes dels distints veïnats de la xarxa, necessitam que el sistema s'explori a sí mateix. Aquest punt és inviable en entorn reals i en xarxes socials real, ja que no disposam d'un mecanisme d'aquest tipus.
- A cada simulació nova s'ha de tornar a calcular el model de regressió.
- Cal disposar prèviament de dades simulades del mètode 3 (en aquest cas).
- **Per tant, aquest mètode 5 té massa dependències amb variables del sistema. Aquest grau de dependència fa que sigui molt inviable implementar-ho per a entorns reals.**

En el pròxim apartat veurem un altre mètode nou que utilitza també la variable del temps mitjà per a fer una estimació del grau de relació dels veïns de la xarxa social, sense utilitzar dependència al sistema.

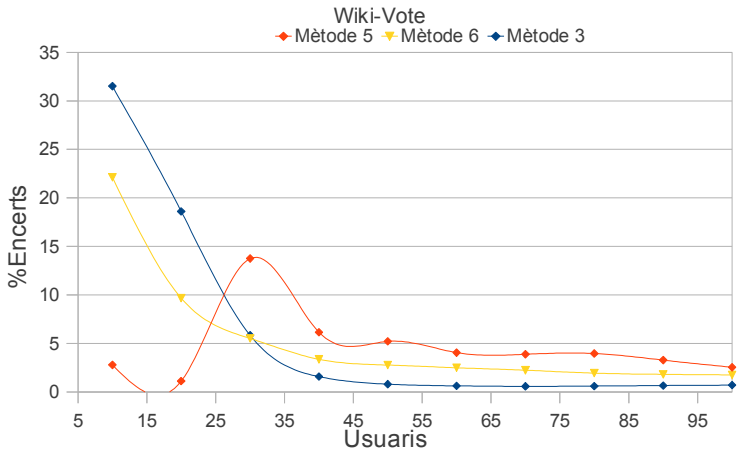
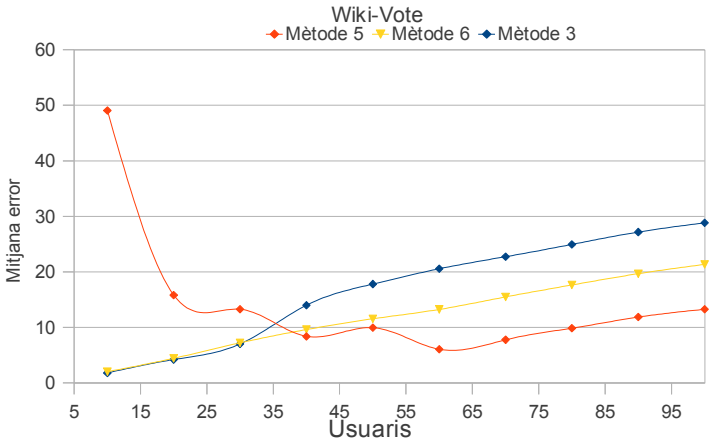
4.6 Modificació del mètode 3: ordenació per temps mitjà.

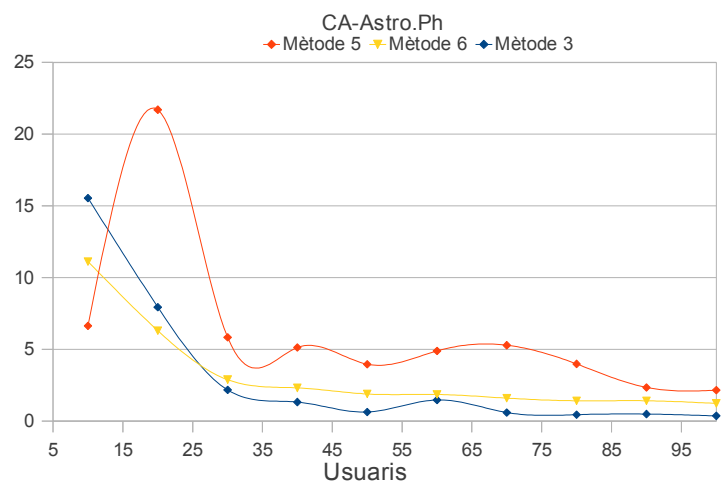
Com hem avançat en l'apartat anterior utilitzarem la variable del temps mitjà per fer una modificació al mètode 3. La intenció és tenir una llista ordenada dels veïnat pel seu temps mitjà en rebre consultes.

En aquest mètode tots els veïnats d'un node son posats en una llista ordenada segons el temps mitjà en que el node rep consultes del veïnat. El veïnat amb menor temps mitjà és situat a la primera posició, mentre que el veïnat amb un temps mitjà més alt és situat a la darrera posició. D'aquesta manera el veïnat amb menor temps mitjà s'assumeix que té una connexió, el segon dues, i així fins el darrer veïnat.

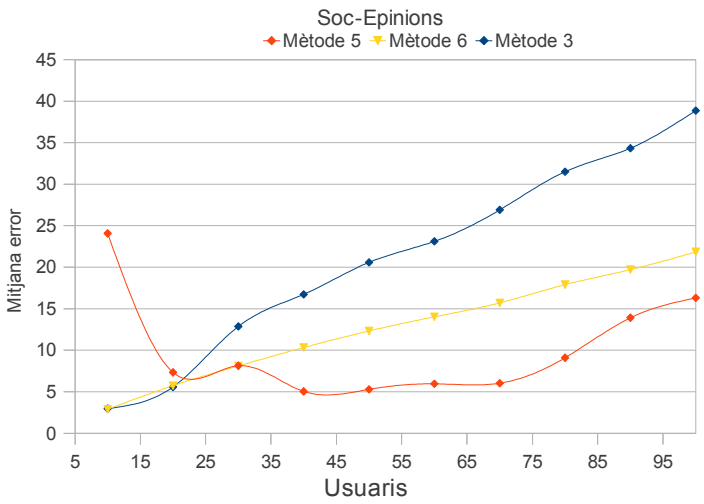
La idea ja s'ha explicada en apartats anteriors on un node amb moltes connexions tendrà un temps mitjà en enviar consultes més alt d'un altre que només en té una.

Fent aquesta implementació (**mètode 6)** i simulant amb les mateixes condicions anteriors a les quatre xarxes escollides obtenim les següent gràfiques, comparades amb el mètode 3:

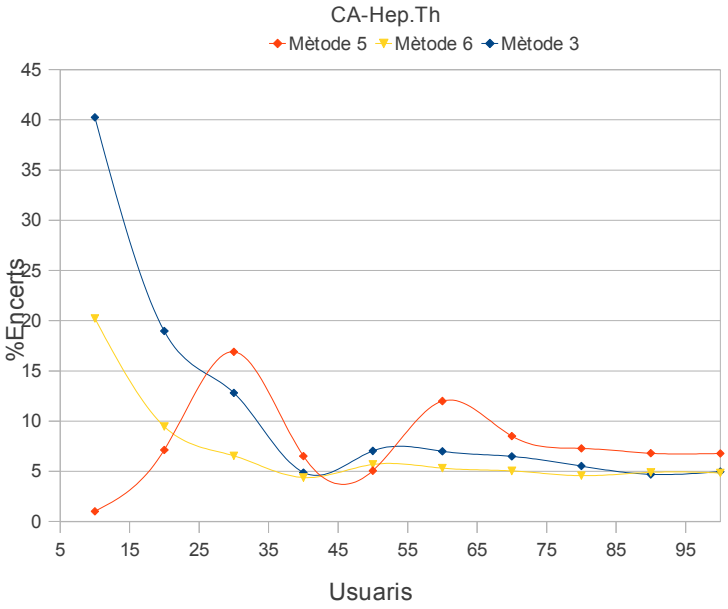
%Encerts	Error mitjà
	
Il·lustració 37: %Encerts mètode 6 (Wiki-Vote)	*Il·lustració 38: Mitjana error mètode 6 (Wiki-Vote)*



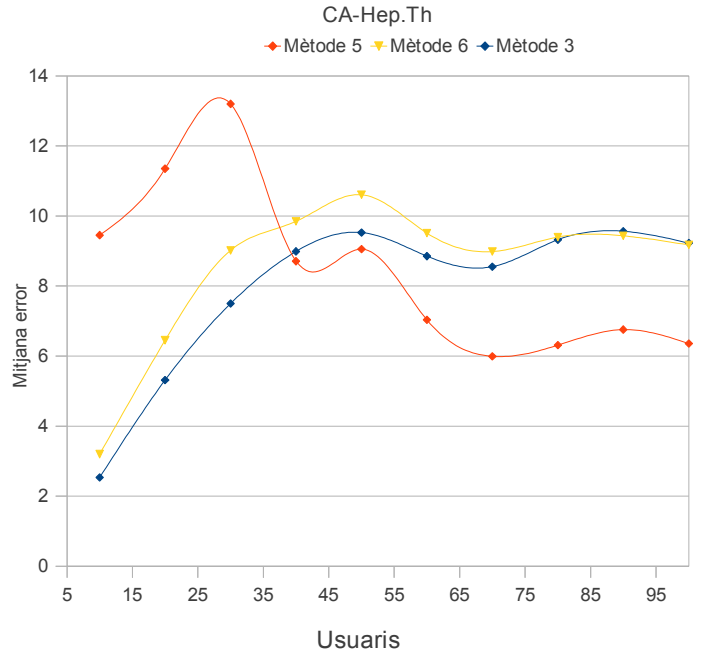
Il·lustració 39: %Encerts mètode 6 (Soc-Epinions)



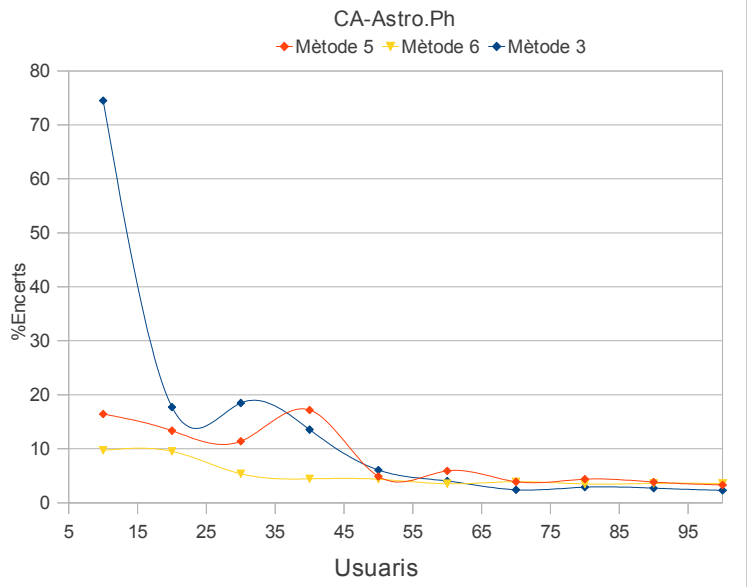
Il·lustració 40: Mitjana error mètode 6 (Soc-Epinions)



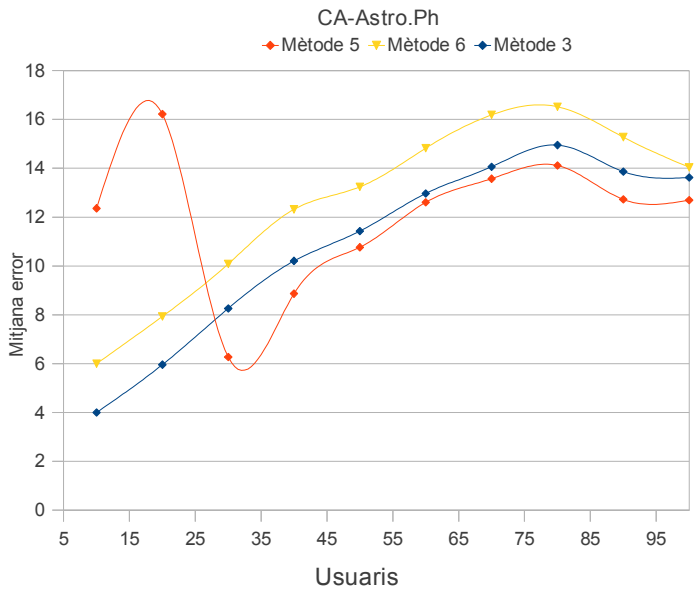
Il·lustració 41: %Encerts mètode 6 (CA-Hep.Th)



Il·lustració 42: Mitjana error mètode 6 (CA-Hep.Th)



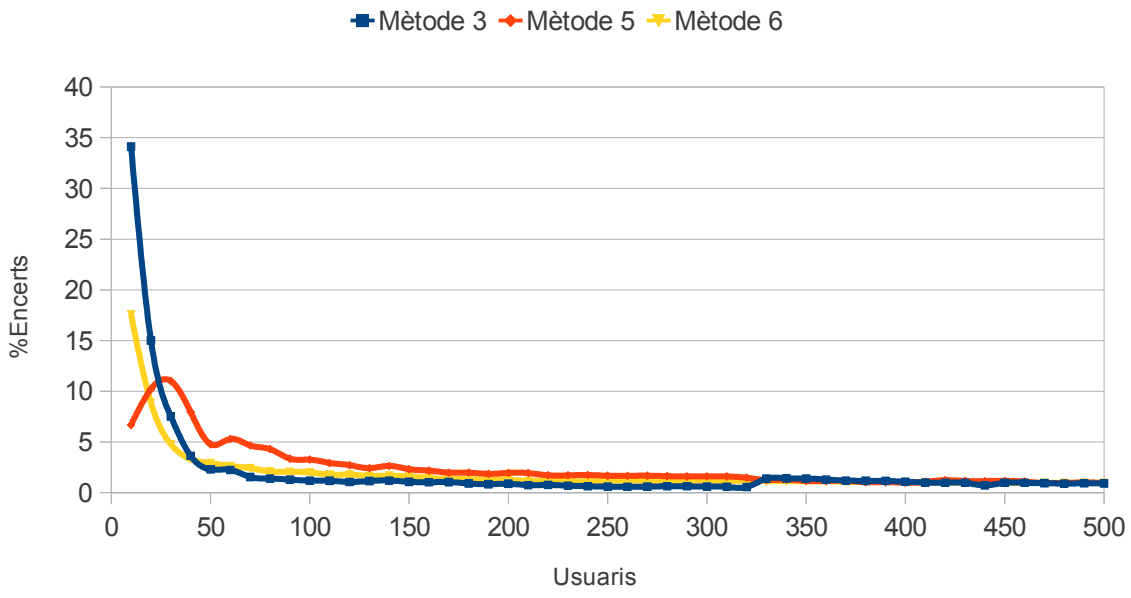
Il·lustració 43: %Encerts mètode 6 (CA-Astro.Ph)



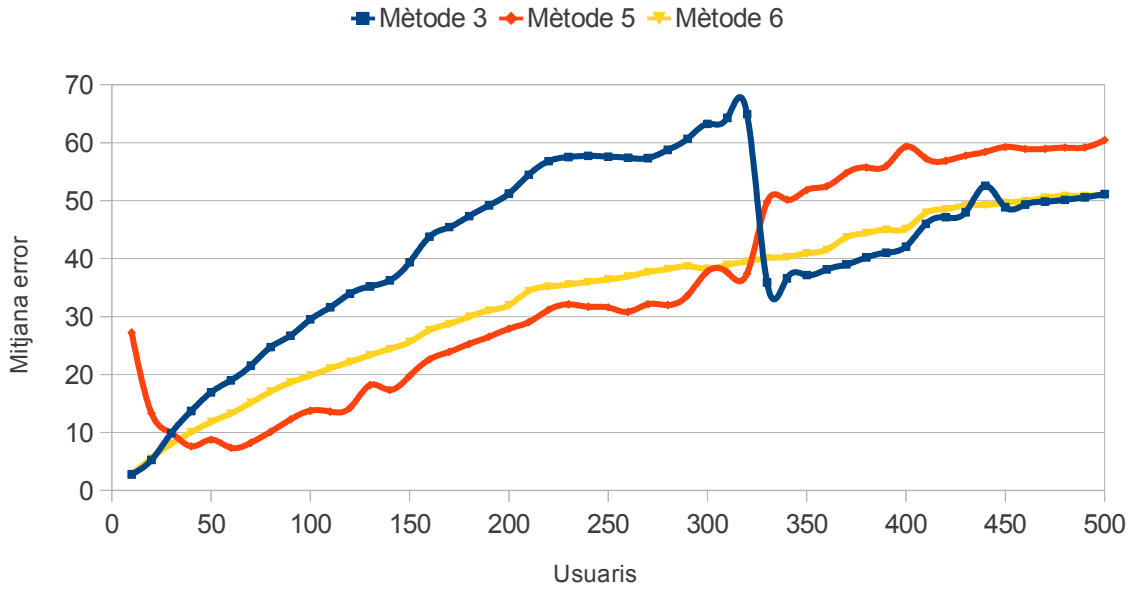
Il·lustració 44: Mitjana error mètode 6 (CA-Astro.Ph)

A les quatre gràfiques és pot veure com per a un nombre relativament petit d'usuaris o nodes de la xarxa, el mètode 6 es comporta de manera semblant al mètode 3.

Per un nombre de nodes més elevat el comportament del mètode 6 és millor que el mètode 3, especialment per aquelles xarxes que tenen moltes relacions entre els nodes (*Wiki-Vote* i *Soc-Epinions*). El guany no és tan pronunciat per a dir que el mètode 6 és globalment millor, però si que es pot apreciar que no baixa mai del 1%, cosa que amb el mètode 3 (i els altres mètode també) si que ocorre.



Il·lustració 45: %Encerts globals (mètode 3,5 i 6)



Il·lustració 46: Mitjana error global

4.7 Quadre resultant globals dels mètodes i conclusions de les simulacions

A partir de les simulacions i dades generades en el projecte podem confeccions la següent taula general i en podem extreure les següents conclusions dels nous mètodes implementats:

- El mètode 5 millora el rendiment dels altres mètodes, per a quantitats de nodes elevats >20. Per pocs nodes no és tan útil com el 3 per exemple (veure il·lustració 45)
- El mètode 5 és requereix una complexitat mot major que el altres (inclòs el 6) i per això es fa difícil per a entorns reals.
- El mètode 6 millora el rendiment també per un nombre d'usuaris mitjà o alt, encara que amb poc nodes dóna un resultat satisfactori.
- El mètode 6 és implementable en situacions reals.

Taula 7: Quadre de resultats globals

Mètodes	**Entre 10 i 20 nodes**		**Entre 20 i 50 nodes**		**Entre 50 i 100 nodes**		**Entre 100 i 200 nodes**		**Entre 200 i 500 nodes**	
	%	Desviació	%	Desviació	%	Desviació	%	Desviació	%	Desviació
Mètode 1	11,6	6,4	3,77	11,7	2,39	18,22	1,8	27,31	1,71	43,17
Mètode 2	10,1	15,38	3,05	23,28	2,00	33,7	1,4	46,29	1,22	64
Mètode 3	20,05	4,6	4,45	13,9	1,51	24,1	1,02	41,97	0,88	51,9
Mètode 4	16,59	5,16	4,19	14,41	1,52	24,9	0,93	42,24	0,78	55,21
Mètode 5	**9,31**	**16,87**	**7,13**	**8,89**	**3,94**	**11,00**	**2,26**	**22,10**	**1,25**	**50,5**
Mètode 6	**11,13**	**4,8**	**3,69**	**10,28**	**2,23**	**16,94**	**1,51**	**26,9**	**1,05**	**44,9**

4.8 Propostes de futur

Deixam remarcades les següents propostes relacionades amb aquest projecte per a que es puguin anar desenvolupant en un futur:

- Projecte de mineria de dades per extreure coneixement de l'elevada quantitat de dades generades pel simulador.
- Desenvolupar un simulador *multi-thread* que augmenti el seu propi rendiment, de cara a simulacions molt fortes.
- Proporcionar més realisme al simulador fent que les comunicacions entre els nodes sigui via xarxa (http, sockets, etc.) amb la possibilitat de distribuir-ho a molts de nodes de la xarxa.
- Incorporació al projecte de llibreries gràfiques per a visualitzar les dades de manera ràpida i intuïtiva.

5 Conclusions

Al llarg del projecte s'ha vist clar que protegir la privacitat dels usuaris que utilitzen els cercadors web, enfront a possibles perfilats per part d'aquests cercadors, és un punt important i, cada pic més. De totes maneres s'ha d'intentar aconseguir un compromís entre, per una banda el nivell de privacitat cercat i per altre el nivell de qualitat de la cerca ja que quan més informació es té de l'usuari, més acurades podran ser les cerques.

S'ha descrit el protocol proposat per Alexandre Viejo i Jordi Castellà-Roca en la seva publicació "*Using social networks to disort users' profiles generated by web search engines*", per protegir aquestes cerques i els possibles perfilats. En aquesta proposta s'especifiquen unes pautes i algorismes que intenten distribuir les consultes enviades al motor de cerca web entre tots els usuaris associats a una xarxa social, evitant altres solucions que no asseguren la privacitat (proxies, etc.) i garantint que el sistema no pugui ser utilitzat de forma malintencionada per realitzar l'enviament de consultes il·legals o bé perfilar altres usuaris de la mateixa xarxa social.

El protocol en que es base aquest projecte té dues funcions claus per a poder distribuir uniformement les consultes que es van generant en tots els usuaris de la xarxa al mateix temps que intentar premiar el usuaris honests i castigar el usuaris amb comportament egoista (ens referim a aquells usuaris que utilitzen el protocol únicament per resoldre les seves consultes sense col·laborar en la resolució de les consultes dels usuaris veïnats). Aquestes dues funcions hem vist que calculaven respectivament el nivell d'exposició de l'usuari candidat a enviar la consulta i l'estimació del nivell d'egoisme dels usuaris. S'ha vist la necessitat que per a calcular el nivell d'exposició de l'usuari candidat es necessita fer una estimació del quantitat de nodes que té el candidat. Aquesta dada no es pot saber sinó que s'ha d'aproximar.

En el projecte s'ha estudiat els comportaments dels mètodes proposats per Alexandre Viejo i Jordi Castellà-Roca al seu article i s'ha vist la necessitat de millorar l'eficiència d'aquests mètodes. Aquest ha estat un dels nostres principals objectius.

Com s'ha dit abans un dels principals objectius del projecte és l'estudi dels mecanismes d'estimació de la mida de la xarxa i la proposta de nous mètodes que en millorin el seu rendiment. Per fer-ho, però necessitam disposar d'un simulador adient que ens pugi executar el comportament del protocol descrit, obtenir-ne les dades, analitzar-les, proposar nous mètodes i contrastar-ho amb els que ja coneixem. Resumim breument els objectius del projecte:

- Implementar un simulador capaç de executar el protocol descrit per Alexandre Viejo i Jordi Castellà-Roca i obtenir-ne dades especialment referides als mètodes d'estimació dels nodes de la xarxa.
- Estudiar el comportament d'aquests mètodes en base a les dades obtingudes.

- Proposar nous mètodes que millorin el percentatge d'encerts dels mètodes ja proposats.
- Estudiar i comparar aquest mètodes amb els ja implementats

Un cop finalitzat el disseny i la implementació del simulador (amb dos models de solucions, una gràfica GUI i intuïtiva i una altra més senzilla però ràpida per obtenir gran quantitat de dades), s'ha passat a la fase d'estudiar les xarxes i el comportament dels mètodes ja descrits.

Gràcies en aquests estudis s'ha entès un concepte molt important en aquests sistemes: el perfil asimptòtic de les xarxes socials que segueixen una distribució *power-law* on hi ha molts d'usuaris/nodes que tenen pocs veïnats i pocs nodes que tenen molts de veïnats.

Gràcies també a l'estudi d'aquestes dades s'ha vist que el rendiment dels mètodes d'estimació proposats al protocol disminueixen com més usuaris hi intervenen a la xarxa. I tan important com la quantitat d'usuaris que té la xarxa com la quantitat de **relacions**.

Estudiant les dades obtingudes a partir del simulador implementat s'ha vist que hi ha un patró a tenir en compte: com més alt sigui el **temps mitjà** en que un node rep consultes del seu veïnat, és probable que aquest veïnat tengui més connexions amb altres nodes, que amb temps mitjans menors. Aquest fet ens ha fet pensar en la possibilitat d'inferir un mètode d'estimació de la mida d'un node en base a aquesta variable. Utilitzant aquest fet, s'ha aproximat aquest comportament a una recta utilitzant mètodes de regressió lineal. Amb aquestes condicions s'ha simulat totes les dades i s'ha vist que en general aquest nou mètode (**mètode 5**) millora el rendiment del millor mètode estudiat (mètode 3). Però s'ha vist clar també que aquest mètode té massa dependència amb el conjunt del sistema per dues raons: per calcular les rectes de regressió és necessita prèviament disposar de dades generades, és a dir, necessitam que la xarxa s'explori a sí mateixa. Aquest comportament no es adient implementar-ho en el món real.

Utilitzant la variable analitzada del temps mitjà en que un usuari rep consultes del seus veïnats, s'ha proposat un altre mètode (**mètode 6**) on tots els veïnats d'un node son posats en una llista ordenada segons aquest temps mitjà. El veïnat amb menor temps mitjà és situat a la primera posició, mentre que el veïnat amb un temps mitjà més alt és situat a la darrera posició. D'aquesta manera el veïnat amb menor temps mitjà s'assumeix que té una connexió, i així successivament. Amb els resultats de les simulacions s'ha vist com amb aquest mètode es millorava el rendiment (comparant-ho amb el mètode 3. A més a més s'ha vist que el mètode és més pràctic que el mètode basat en regressions i es pot dur en entorns reals.

Per altra banda, al llarg del desenvolupament del projecte, especialment en la fase d'anàlisi de dades, en diverses ocasions ha sortir la idea de intentar fer (per a treballs futurs) algun projecte de mineria de dades per tal d'extreure coneixement a partir del gran volum de dades generat pel simulador.

6 Glossari

Ant: eina utilitzada a l'hora de crear projectes Java per a la realització de tasques mecàniques i repetitives, normalment durant la fase de compilació i construcció de l'aplicació.

Client-servidor: Arquitectura del programari consistent en que un programa (client) realitza peticions a un altre programa (servidor) que li serveix la resposta.

Criptografia: Ciència que estudia les tècniques matemàtiques relacionades amb els diferents aspectes de la seguretat de la informació.

Entitat: És una representació abstracta i conceptual d'una dada i la seva estructura.

Escalabilitat: És la propietat d'un sistema, xarxa o procés, que indica l'habilitat per menejar el creixement continuu del treball de manera fluida i estar en preparació per esdevenir més gran sense pèrdua de qualitats en els serveis oferters.

Framework: és una estructura de suport definida a través de la qual un altre projecte de software pot ser organitzat i desenvolupat. Sol incloure suport de programes, biblioteques, llibreries per ajudar a desenvolupar i ajuntar els diferents components d'un projecte.

Herència: és la propietat que permet als objectes ser creats a partir d'altres ja existents, obtenint característiques (mètodes i propietats) similars als ja existents. És la relació entre una classe general i una altre classe més específica.

Heterogeneïtat: Propietat d'un sistema distribuït que indica que està format per una varietat de diferents xarxes, sistemes operatius, llenguatges de programació o maquinari de l'ordinador o del dispositiu.

Indexació: acció de registrar ordenadament informació per elaborar un índex amb la finalitat d'obtenir resultats de forma més ràpida i rellevant en el moment de realitzar una consulta.

Interfície: Punt d'interacció i comunicació entre un ordenador i una altra entitat.

JDK: Acrònim de Java Development Kit. Intèrpret i entorn per a desenvolupaments Java.

JDBC: Capa de connexió amb bases de dades relacionals des de Java. Ofereix independència lògica respecte a la base de dades escollida.

Junit: És un conjunt de classes i llibreries (framework) que permet realitzar l'execució de classes i programes Java de manera controlada per a poder avaluar si el funcionament de cada un del mètodes de la classe es comporta com s'espera (prove unitàries).

Llibreria: Conjunt de classes reutilitzables i relacionades les unes amb les altres.

Log: Zona del sistema on s'anoten les incidències que van ocorrent. Emmagatzema informació del funcionament i proporciona informació per a la detecció de problemes d'aplicacions i sistemes

Lògica de negoci: és la part d'un sistema (en arquitectura de software) que s'encarrega de les tasques relacionades amb els processos de negoci i tota casta de processament que es realitza

darrera de l'aplicació visible a l'usuari.

Meta-Inf: És un directori que presenten les aplicacions desenvolupades en Java destinat a contenir la meta informació sobre els fitxers continguts a l'aplicació.

Motor de cerca: Programa informàtic dissenyat per ajudar a trobar informació emmagatzemada a un sistema informàtic com és una xarxa, Internet, o un ordinador personal. El motor de cerca permet demanar contingut que satisfaci un criteri determinat (típicament que contingui una paraula o frase donada) i retorna una llista de referències que compleixen aquest criteri

Serialització: procés de codificació d'un objecte e un mitjà d'emmagatzemament amb la finalitat de ser transmés a través d'una conneció de xarxa com un serie de bytes .

Sistema distribuït: Col·lecció d'ordinadors autònoms enllaçats per una xarxa d'ordinadors i suportats per un programari que fa que la col·lecció actuï com un servei integrat.

UI (User Interface): És un mitjà amb el qual l'usuari es pot comunicar amb una màquina, un equip o computadora, i compren tots els punts de contacte entre usuari i equip, normalment fàcils d'entendre
i accionar.

UML (Unified Modeling Language): És un llenguatge de modelat de sistemes de software utilitzat de manera gràfica per visualitzar, especificar, construir i documentar un sistema.

7 Referències

[1] Kristen Purcell (2011). Pew Research Center's Internet & American Life Project. http://pewinternet.org/Reports/2011/Search-and-email.aspx

[2] Myaeng, S. H. and Korfhage, R. R. (1986). Towards an intelligent and personalized retrieval system. In Proceedings of the ACM SIGART international symposium on Methodologies for intelligent systems, pp 121-129, Knoxville, Tennessee, United States. ACM Press. ISBN:0-89791-206-3.

[3] K. Sugiyama, K. Hatano, M. Yoshikawa, Adaptive web search based on user profile constructed without any effort from users, in: Proc. of the 13th international conference on World Wide Web, 2004, pp. 675–684.

[4]F. Qiu, J. Cho, Automatic identification of user interest for personalized search, in: Proc. of the 15th international conference on World Wide Web, 2006, pp. 727–736.

[5] J. Teevan, S. Dumais, E. Horvitz, Personalizing search via automated analysis of interests and activities, in: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, 2005, pp. 449–456.

[6] R. Cooley, B. Mobasher, and J. Srivastava. Data Preparation for Mining World Wide Web Browsing Patterns. Knowledge and Information Systems, pp 5–32, 1999.

[7] X. Fu, J. Budzik, and K. J. Hammond. Mining Navigation History for Recommendation. In Proc. of the 5th International Conference on Intelligent User Interfaces (IUI 2000), pp 06–112, 2000.

[8] M. Spiliopoulou and L. Faulstich. WUM–A Tool for WWW Utilization Analysis. In Proc. of the International Workshop on the World Wide Web and Databases (WebDB'98), pp 184–203, 1998

[9] J. Wang, Z. Chen, L. Tao, W.-Y. Ma, and L. Wenyin. Ranking User's Relevance to a Topic through Link Analysis on Web Logs. In Proc. of the 4th ACM CIKM International Workshop on Web Information and Data Management (WIDM'02), pp 49–54, 2002.

[10]Xuehua Shen, Bin Tan, ChengXiang Zhai. Privacy Protection in Personalized Search. Department of Computer Science. University of Illinois at Urbana-Champaign pp 7.

[11] https://www.torproject.org/

[12] http://en.wikipedia.org/wiki/Private_information_retrieval

[13] Alexandre Viejo, Jordi Castellà-Roca. "Using social networks to distort users' profiles generated by web search engines", Computer Networks 54(2010) pp:1343–1357

[14] AOL search data scandal, http://en.wikipedia.org/wiki/AOL_search_data_leak

[15] P. A. Chirita, W. Nejdl, R. Paiu, and C. Kohlschütter. Using ODP metadata to personalize search. In Proc. of the 28th Annual Intel ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR'05), pp 178– 185, Salvador, Brazil, 2005.

[16] H. rae Kim and P. K. Chan. Personalized ranking of search results with learned user interest hierarchies from bookmarks. In Proc. of the 7th WEBKDD workshop on Knowledge Discovery from the Web (WEBKDD'05), pp 32–43, Chicago, Illinois, USA, 2005.

[17] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In Proc. of the 2005 ACM CIKM Int'l Conf. on Information and Knowledge Management (CIKM'05), pp 824–831, 2005.

[18] H. rae Kim and P. K. Chan. Personalized ranking of search results with learned user interest hierarchies from bookmarks. In Proc. of the 7th WEBKDD workshop on Knowledge Discovery from the Web (WEBKDD'05), pp 32–43, Chicago, Illinois, USA, 2005

[19] S. Staab, P. Domingos, P. Mika, J. Golbeck, L. Ding, T.W. Finin, A. Joshi, A. Nowak, R.R. Vallacher, Social networks applied, IEEE Intelligent Systems 20 (1) (2005) pp:80–93.

[20] R. Ashri, S.D. Ramchurn, J. Sabater, M. Luck, N.R. Jennings, Trust evaluation through relationship analysis, 4th International Joint Conference on Autonomous Agents and Multiagent

Systems, ACM, 2005. pp. 1005–1011.

[21] J. Sabater-Mir, Towards the next generation of computational trust and reputation models, Modeling Decisions for Artificial Intelligence-MDAI, LNCS 3885, Springer-Verlag, 2006. pp. 19–21.

8 Annexos

8.1 Base de dades

```
CREATE TABLE `simulacio` (
  `sim_id` int(11) NOT NULL AUTO_INCREMENT,
  `sim_nom` varchar(255) NOT NULL,
  `sim_file` varchar(255) DEFAULT NULL,
  `sim_users` bigint(20) DEFAULT NULL,
  `sim_relations` bigint(20) DEFAULT NULL,
  `sim_factor` float DEFAULT NULL,
  `sim_level` float DEFAULT NULL,
  `sim_method` varchar(255) DEFAULT NULL,
  `sim_queries` bigint(20) DEFAULT NULL,
  `sim_performance` float DEFAULT NULL,
  `sim_desviation` float DEFAULT NULL,
  `sim_max` float DEFAULT NULL,
  `sim_min` float DEFAULT NULL,
  `sim_created` datetime DEFAULT NULL,
  PRIMARY KEY (`sim_id`)
);

CREATE TABLE `resultat` (
  `res_id` int(11) NOT NULL AUTO_INCREMENT,
  `res_simid` int(11) DEFAULT NULL,
  `res_user` varchar(255) DEFAULT NULL,
  `res_neighbour` varchar(255) DEFAULT NULL,
  `res_qatone` bigint(20) DEFAULT NULL,
  `res_qstone` bigint(20) DEFAULT NULL,
  `res_qafrne` bigint(20) DEFAULT NULL,
  `res_qsfrne` bigint(20) DEFAULT NULL,
  `res_intents` int(11) DEFAULT NULL,
  `res_encerts` int(11) DEFAULT NULL,
  `res_tempsreb` float DEFAULT NULL,
  `res_tempsenv` float DEFAULT NULL,
  `res_nodes` int(11) DEFAULT NULL,
  `res_veinats` int(11) DEFAULT NULL,
  `res_estimat` float DEFAULT NULL,
  PRIMARY KEY (`res_id`)
) ;
```

8.2 Consultes

```
--Rendiment
select sim_queries,sim_method,avg(sim_performance),avg(abs(res_veinats-res_estimat))
from simulacio
inner join resultat on res_simid = sim_id
where sim_method in ('3','5','6') and
      sim_users=50
group by sim_method,sim_queries
order by sim_method,sim_queries;

--Densitat
select a.veinats, count(a.res_user) as nodes
from(
  select res_user, count(res_neighbour) veinats
```

```
  from simulacio
  inner join resultat on res_simid=sim_id
  where sim_users=500 and sim_queries=100 and sim_file='CA-HepTh.txt'
  group by res_user
) a
group by a.veinats
order by a.veinats;
--Consultes rebudes
select res_veinats, avg(res_qsfrne)
from simulacio
inner join resultat on res_simid=sim_id
where sim_users =500 and sim_queries=100 and sim_file='CA-HepTh.txt'
group by res_veinats
order by res_veinats;

--Consultes enviades
select res_nodes, avg(res_qstone)
from simulacio
inner join resultat on res_simid=sim_id
where sim_users =500 and sim_queries=100 and sim_file='soc-Epinions.txt'
group by res_nodes
order by res_nodes;

--Temps mitja en rebre
select res_veinats, avg(res_tempsreb)
from simulacio
inner join resultat on res_simid=sim_id
where sim_users =500 and sim_queries=100 and sim_file='CA-HepTh.txt' and sim_method='4'
group by res_veinats
order by res_veinats;
```