,pdfcreator=HAL,pdfproducer=PDFLaTeX,pdfsubject=Computer Science [cs]/Other [cs.OH]

Integrating File Popularity and Peer Generosity in Proximity Measure for Semantic-based Overlays

Yann Busnel, Anne-Marie Kermarrec

To cite this version:

Yann Busnel, Anne-Marie Kermarrec. Integrating File Popularity and Peer Generosity in Proximity Measure for Semantic-based Overlays. [Research Report] PI 1756, 2005. inria-00000502

HAL Id: inria-00000502

https://inria.hal.science/inria-00000502

Submitted on 25 Oct 2005

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

I R I S A
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1756

INTEGRATING FILE POPULARITY AND PEER GENEROSITY IN PROXIMITY MEASURE FOR SEMANTIC-BASED OVERLAYS

YANN BUSNEL AND ANNE-MARIE KERMARREC

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES
Campus de Beaulieu – 35042 Rennes Cedex – France
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

Integrating File Popularity and Peer Generosity in Proximity Measure for Semantic-based Overlays

Yann Busnel * and Anne-Marie Kermarrec **

Systèmes numériques
Projet Paris

Publication interne n ° 1756 — Octobre 2005 — 23 pages

Abstract: Peer-to-peer file sharing systems are now at the origin of most of Internet traffic. Improving the performance of such systems has generated a lot of interest both in industry and academia. More specifically, many approaches focus on the improvement of the query mechanism in such systems. In a peer-to-peer system, peers are connected to a subset of other peers with which they can communicate. Each peer maintains a cache and makes available its contents to the rest of the system. Connecting peers sharing similar interest in the context of a given application has recently been identified as a sound basis to improve the search efficiency.

However, capturing such interest-based (or semantic) proximity patterns is a difficult task. Most of current approaches measure this proximity between peers as the overlap between their cache contents. Given the well-known popularity patterns of peer-to-peer file sharing systems, the overlap between cache contents of two peers may not reflect accurately their semantic proximity. More specifically, this measure depends upon peer generosity and file popularity.

In this paper we propose a refined proximity measure taking into account these factors. We evaluated the proposed solution by simulation against a real peer-to-peer system (eDonkey) workload and results show the effectiveness of the proposed approach. While peers generosity can easily be computed locally, file popularity may require a global knowledge of the system. We also propose in this paper an epidemic algorithm to compute in a fully decentralised fashion an estimation of files popularity.

Key-words: Distributed system, Peer-to-peer network, gossip-based protocol, interest-based proximity measure, semantic profile

(Résumé : tsvp)

* Yann.Busnel@irisa.fr
** Anne-Marie.Kermarrec@irisa.fr

Centre National de la Recherche Scientifique
(UMR 6074) Université de Rennes 1 – Insa de Rennes

Institut National de Recherche en Informatique
et en Automatique – unité de recherche de Rennes

Prise en compte de la popularité des fichiers et de la générosité des nœuds dans une mesure de proximité d'intérêt pour les réseau pair-à-pair

Résumé : Ce rapport présente une mesure de proximitšémantique permettant de prendre en compte les effets de bords induits par la générosité des nœuds et la popularité des fichiers d'un système de partage de fichiers. Cette mesure est introduite dans un protocôle épidémique permettant la comparaison des résultats obtenus.

De plus, ce rapport introduit une approche épidémique d'évaluation locale de la popularité d'un fichier.

Mots clés : Systmes distribus, rseaux pair--pair, protocoles pidmiques, profil smantique, proximit d'intrt

1 Introduction and background

Peer-to-peer file sharing systems Peer-to-peer (P2P) overlay networks have recently proved to be efficient to support a large spectrum of large-scale distributed applications. P2P overlay networks and their applications have generated a lot of interest in the research community in the past five years spanning from the overlays networks themselves [9, 18, 21] to streaming, archival, voice on IP applications, etc. [5, 7, 11]. However, the main P2P applications deployed on Internet today are the P2P file sharing systems. They represent now the highest consumers of Internet bandwidth [19].

File sharing systems may be implemented over unstructured, structured or hierarchical overlay networks. The query mechanism is dependent upon the structure of the underlying network. In unstructured overlays, such as for example the early version of Gnutella [2], the query approach is implemented by flooding the system. On the other hand, hierarchical overlays are composed of a set of super peers to which clients are connected. Super peers are in charge of indexing the clients cache contents and redirect the requests towards relevant peers. KaZaA [3] and eDonkey [1] networks rely on such hierarchical models. Structured overlays provide an efficient exact-search mechanism by providing distributed hash table (DHT) functionality [4]. Many approaches have been proposed to improve search mechanisms in such file sharing systems by optimising the replication strategies [17], using random walk instead of flooding to improve the load-balancing [9] or combining Gnutella-like systems with some structured overlays [6, 16].

Semantic proximity While generic P2P infrastructures have been optimised to take into account physical locality [8] early on, some recent work relies on capturing and exploiting other forms of proximity such as interest-based, or semantic, proximity[1]. Peers in a file sharing system may exhibit similar download patterns and/or have similar cache contents. These similarities may be exploited to define a semantic proximity measure and to improve the cost and efficiency of the query mechanism. These approaches are motivated by the fact that semantically related peers are more likely to be useful to each other than peers picked at random [22].

Almost all query mechanisms, whether they rely on flooding, DHT, or super-peers based systems, may be improved by identifying semantically related peers and querying those peers (called semantic neighbours in the remaining of this paper)

[1] We stick to the term semantic proximity in the remaining of this paper.

first. If the request is not satisfied, a second phase is launched using the standard query mechanism.

Detecting semantic relationships Various approaches can be used to capture semantic proximity between peers. One way is to connect "similar" peers according to a predefined metric in separated overlays. Each overlay is explicitly identified by the type of documents or a predefined ontology (semantic classification) [10]. Unfortunately, defining a precise set of classification items is not an easy task and such an ontology may need to change over time to reflect the change of semantic profiles over time. At the other end of the spectrum, another approach is to add some semantic shortcuts (*i.e.* additional links) between peers that share some interest [13, 20]. These links are created dynamically between peers, based on the set of most recent downloads for instance. Such a mechanism is very reactive to evolving download patterns. However, the non-intrusive nature of this approach does not allow to exploit further available information such as the overlap between caches for example.

Detecting and measuring semantic proximity is a difficult task. In [12], an analysis of clustering in peer-to-peer file sharing system traces has shown the existence of semantic clustering, based on the analysis of peers contents overlap. In addition to recent download patterns, the overlap between cache contents may be used to measure the semantic proximity between peers [23]. In [22], an evaluation of several strategies to capture semantic proximity has been conducted. More specifically, simple strategies based on past requests behaviour are compared. One observation is that assessing semantic proximity this way may lead to biased measurements. This is mostly due to the presence of generous peers and popular files which tend to hide genuine locality. Tacking into account these factors is the main goal of this paper.

Gossip-based protocols for tracking semantic proximity The approach presented in [23] relies on a gossip-based protocol to explicitly detect proximity between peers. As mentioned before, the proximity is here measured as the overlap between peers cache contents. The underlying overlay is then updated accordingly. The gossip-based protocol is composed of two layers. The bottom one ensures connectivity [15] while the top one is used to improve semantic search. We will use this gossip-based approach as a basis in this paper to evaluate the proposed solution.

Contribution In this paper, we propose a refined semantic proximity measure capturing peer generosity and file popularity in addition to overlap between cache

(a) active thread

at each cycle, do
$q \leftarrow$ GETNEIGHBOUR()
send κ_p to q
$\kappa_q \leftarrow$ receive(q)
$\kappa_p \leftarrow$ UPDATEVIEW(κ_p, κ_q)

(b) passive thread

do forever
$\kappa_q \leftarrow$ receive(*)
send κ_p to sender(κ_q)
$\kappa_p \leftarrow$ UPDATEVIEW(κ_p, κ_q)

Figure 1: Gossip-based protocol executed by peer p

contents. Quite a few approaches have been proposed to use semantic proximity but to the best of our knowledge not to refine the semantic measure to take into account those issues. We use a gossip-based overlay similar to the one presented in [23]. We also propose a decentralised gossip-based algorithm to assess file popularity. We evaluate the improvement of our metric using a eDonkey 2000 [1] network trace obtained in November 2003 [12].

This paper is organised as follows: in Section 2, we present the design rationale of the proposed approach; in Section 3, we define the semantic proximity measure; in Section 4, we present the evaluation of the proposed measure; in Section 5, we introduce a decentralised protocol to estimate file popularity, and finally conclude in Section 6.

2 System model

Although, the proposed approach may be used in a wide range of peer-to-peer file sharing systems, we present it in the context of an unstructured network. We believe that the context of unstructured P2P overlay networks is particularly relevant given their flexibility. In the rest of the paper, we focus on the capture of semantic relationship between peers.

Using interest-based proximity to improve file-sharing applications require to be able to: capture, evaluate and exploit semantic proximity between peers.

Capture Reorganising the overlay may be done either by adding or switching links. In this paper, we rewire links as in [23], *i.e.* we switch as soon as a better candidate is encountered. Each peer maintains a list of semantic neighbours called its semantic view and denoted κ_i. Neighbours are selected according to a semantic proximity metric. The way a peer comes across potential new

neighbours is due to the implementation of the gossip-based protocol. The algorithm is given in Figure 1. As in [23], a bottom layer ensures connectivity.

Evaluation Each peer needs to evaluate its semantic "distance" to other encountered peers and order them using these measure as a ranking function. The *closest* known peers according to this measure are selected as neighbours. We present this measure in Section 3.

Exploitation Semantic neighbours are first solicited in a search request. Even with a single hop search, studies has shown that for typical P2P file sharing systems workload, we obtain a good hit ratio [13, 22]. If the search based on semantic neighbours fails, the standard one is used.

3 Semantic proximity measure

We introduce a number of proximity measure notations:

- A is the local peer;
- B is a distant peer distinct form A;
- $\xi_A(B)$ is the semantic proximity measure B according to A;
- κ_A (resp. κ_B) is the view of the peer A (resp. B);
- $\sigma_{A,B}$ is the overlap of κ_A and κ_B :
  i.e. $\sigma_{A,B} = \kappa_A \cap \kappa_B$;

For the sake of clarity, starting from a basic semantic measure, we will present successive refinements as the section goes.

3.1 Cache overlap

A basic idea, used in several approaches [13, 14, 23], consists in measuring the semantic proximity between peers as the size of the overlap between their caches. The greater this value, the semantically closer the peers.

We first normalise this value according to A's cache in order to keep the same definition region.

$$\xi_A^1(B) = \frac{|\sigma_{A,B}|}{|\kappa_A|} \tag{1}$$

The more files A and B have in common, the closer to 1 the measure. Note that at this stage, neither the file popularity nor B's generosity are taken into account.

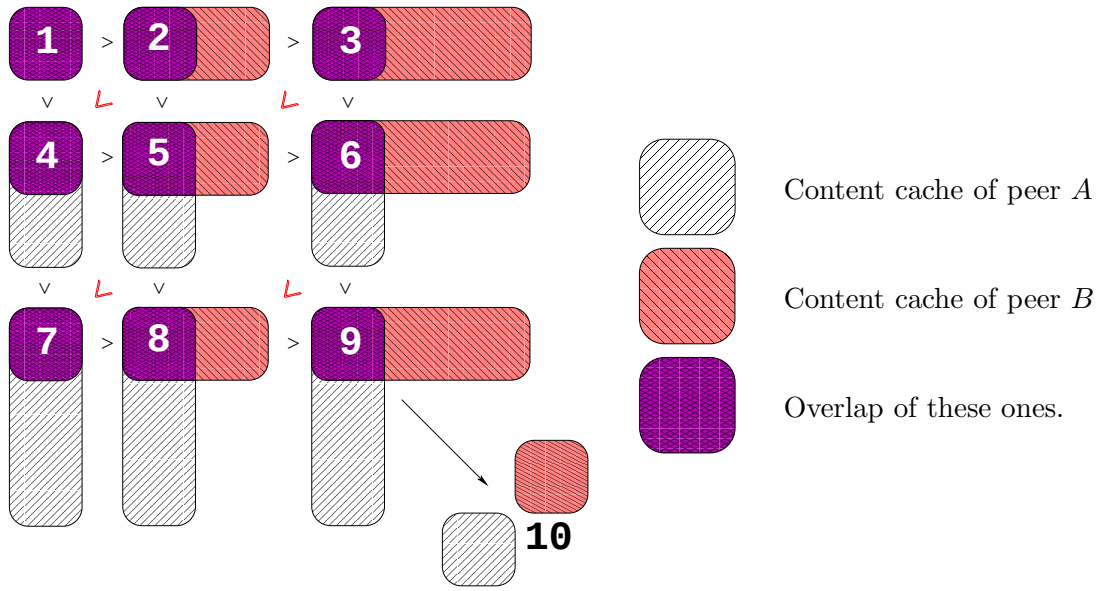


Figure 2: Ordering according to generosity with same overlap size (considered from A's point of view)

3.2 Ignoring generous peers

As introduced in [14], generous peers have a greater probability to share several files with other peers and mask genuine semantic proximity. Therefore, they are more likely to be chosen as a semantic neighbour. Although, this may has a positive effect on performance (obviously a generous peer is able to serve many requests), this may lead to an unbalance in the system. Therefore the semantic proximity measure should take peer generosity into account. Figure 2 presents possible configurations between two peers with varying cache size. In each configuration, the cache size of peers A and B are represented. The overlap $|\sigma_{A,B}|$ is constant for all configurations. These configurations must be ordered where configuration 1 is the best configuration and 10 the worse one. Note that the ordering may not be symmetric depending on which peer measures the semantic proximity.

The semantic proximity measure given in Equation 1 can be adapted as follows to implement partially the requested order. This definition enables to infer the partial

Strategies	*Queries sent to peers*
Random	Chosen at random
Overlap	Selected according to Equation 1
Overlap + Generosity	Selected according to Equation 3
Overlap + Popularity	Selected according to Equation 4
Total (Gen. + Pop.)	selected according to Equation 5

Table 1: Summary

order between configurations 1, 2 and 3 and 1, 4 and 7 as well as 3, 6 and 9 and further, as represented in Figure 2.

$$\begin{cases} \xi_A^2(B) = \alpha \cdot \frac{|\sigma_{A,B}|}{|\kappa_A|} + \beta \cdot \frac{|\sigma_{A,B}|}{|\kappa_B|} \\ \alpha + \beta = 1 \end{cases} \tag{2}$$

To ensure the total order (the order between 2 and 4; 3, 5 and 7; 6 and 8) depicted on this figure, α should be lower than β. The theoretical analysis is provided in Appendix A. We decided to favour configuration 2 over configuration 4: we assume that this order is more relevant than the other for the following reason from A's point of view[2]: In 2, B has a larger cache than the overlap, therefore B has a higher probability to be useful to A in the future. Conversely, 4 represents a setting in which A might be more useful to B. This relation is not symmetric.

To match the previous chosen ordering, a refined definition of the semantic proximity measure is:

$$\begin{cases} \xi_A^2(B) = \alpha \cdot \frac{|\sigma_{A,B}|}{|\kappa_A|} + \beta \cdot \frac{|\sigma_{A,B}|}{|\kappa_B|} \\ \alpha + \beta = 1 \\ \alpha < \beta \end{cases} \tag{3}$$

3.3 Handling file popularity

The study in [14] shows that the popularity of files may impact the semantic proximity. For example, consider two peers, B and C, having a same size overlap with A content cache (*i.e.* $|\sigma_{A,B}| = |\sigma_{A,C}|$) and the same generosity (*i.e.* $|\kappa_B| = |\kappa_C|$). Yet, B and C might not have to be at the same distance from A. For example, B

[2]Note that this order is considered from A's point of view.

may have more popular files in common with A, but C may have more rare files in $\sigma_{A,C}$. Unpopular files (and rare files *a fortiori*) are more representative of a peer's semantic profile. Therefore, the distance between A and C should be greater than the distance between A and B : $\xi_A(B) < \xi_A(C)$.

Equation 3 has to be refined to take this into account. We decided to apply a multiplicative factor to the previous measure, in order to be conservative and have the same definition region. We introduce another notation: τ represents the number of popular files in the set of overlapping files between two peers: $\tau = |\{f | f \in \sigma_{A,B} \wedge f \text{ is popular}\}|$. File popularity is a parameter of the system.

$$\lambda = \left(\frac{|\kappa_B| - \tau}{|\kappa_B|} \right)^{\gamma} \tag{4}$$

where γ is a exponent enabling to give more or less importance to this factor.

Note that the file popularity is defined as the number of replicas of a file. Computing file popularity requires a global knowledge of the system whereas peers only know a subset of the overlay network. We present a mechanism to compute an estimation of the popularity of a file in a distributed way in Section 5.

3.4 Summary

By merging Equation 3 and Equation 4, we obtain a full semantic proximity measure, which can be included in the gossip-based protocol presented in Section 2.

$$\xi_A(B) = |\sigma_{A,B}| \cdot \left(\frac{\alpha}{|\kappa_A|} + \frac{\beta}{|\kappa_B|} \right) \cdot \left(1 - \frac{\tau}{|\kappa_B|} \right)^{\gamma} \tag{5}$$

$$\text{where} \begin{cases} \alpha + \beta = 1 \\ \alpha < \beta \\ \gamma \geq 0 \end{cases}$$

In the remaining of the paper, we compare these variants, summarised in Table 1.

4 Performance evaluation

4.1 Experimental setting

We wrote a discrete-event simulator in which the behaviour of n peers are simulated.

To evaluate the accuracy of the proposed measure, we used a real trace collected from the eDonkey file sharing system [1] in November 2003. This trace has also

been used to evaluate semantic-based systems in [12, 23]. Using the same workload enables to compare directly the results. A set of 12,000 world-wide distributed peers with the files each one shares, is logged in this trace. A total number of 1,100,000 unique files are being collectively shared by these peers. We used that trace as a workload for our simulation the same way as in [23].

We assigned peers randomly to the eDonkey clients of the trace. The simulator maintains the global list of files shared in the system and the popularity of each file[3]. Each client is associated with its list of files according to the real trace and maintains a set of semantic neighbours in its semantic view.

The list of semantic neighbours of each peer is initialised with a set of x random peers. The results presented in this paper are obtained with a set of 20 random peers. For each peer, the simulation consists in executing the active thread of the protocol. At each round, each peer in the system executes the following steps:

1. chooses one of its semantic neighbours, picked randomly in its view;
2. sends to this selected neighbour the content of its own view. The selected neighbour sends back its own view;
3. upon receipt of its neighbour's view, the peer merges the new neighbours known in this exchange with its own view;
4. ranks the peers in the obtained set according to the proximity measure;
5. keeps the x semantically closest neighbours.

This results eventually in a semantic overlay network.

The approaches are compared along two metrics: the hit ratio for rare files and the load on each peer. At each cycle, each peer asks one of the file picked at random among rare files[4] they owned to all its semantic neighbours. A file is considered as a rare file if the number of this file's replicas in the system is lower than a predefined threshold (here, 10 replicas). After a complete round[5], we compute the average hit ratio (for all peers). Rare file are chosen in priority because they are more representative and it is more difficult to locate them than popular ones. To evaluate the impact of generosity, we also compute the load of each peer. This is measured as

[3]We consider for now that the popularity of a file is known *a priori*. We will come back to this issue in Section 5.

[4]Note that the rareness of a file here is globally determined.

[5]A complete round is done when all peers have executed the active thread.

the number of occurrences of a peer in the lists of semantic neighbours (how many peers have chosen this particular peer as a semantic neighbour).

As a basis for comparison, we also use a random search: each peer sends a request for a rare file to x random peers, where x is equal to the number of semantic neighbours. Furthermore, we compare all results with the simulation in which the overlay network is randomly totally rewired at each cycle.

Below, we summarise the simulation parameters:

- Number of peers in the system : 11,291 peers (without *free-riders*)
- Number of files in the system : 1,268,536 files
- Number of cycles : 50
- Size of semantic view : 20 peers
- Popularity threshold : 10 replicas
  File with less than 10 replicas in the system is considered as a rare one.
- Minimum cache content : 20 files
  Peer with less than 20 files in their cache are not integrated in the simulation in order to get rid of *free-riders*.

4.2 Hit ratio results

Figure 3 presents the average hit ratio depending on the cycle of the gossip-based protocol for the four considered strategies.

We observe that the hit ratio of the random approach does not exceed 1%. We observe that only after 5 cycles of the gossip-based protocol, all other versions of the protocol largely outperform the random one. The first dash plot represents the cache overlap hit ratio (corresponding to Equation 1). The second dash plot over the overlap one, represents the hit ratio corresponding to Equation 4 where popularity of requested files is considered while choosing semantic neighbours. In that case, the hit ratio reaches 21% (instead of 17% in the simple overlap case). When peer generosity is taken into account (Equation 3), the average hit rate decreases slightly as expected. The goal of the strategy is more to balance the load in the network than to improve the hit ratio performance. The last curve, call "Total", represents the average value when all factors are integrated in the semantic proximity measure. Results are close to the ones obtained with generosity. We observe that the two effects (generosity and popularity) have actually opposite impacts.

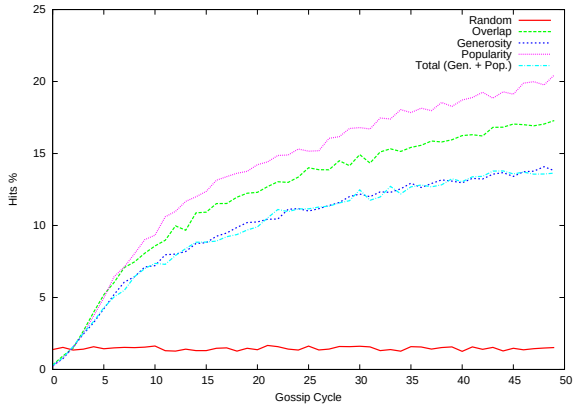


Figure 3: Hit ratio

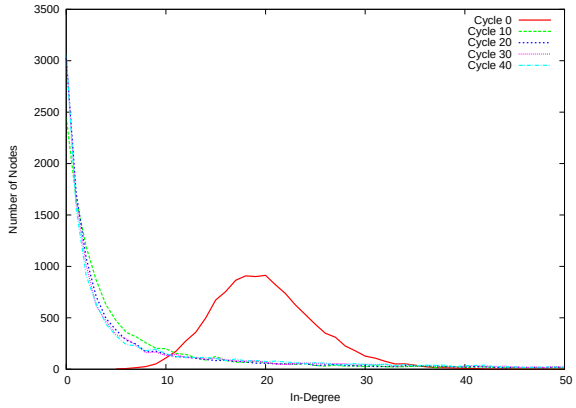
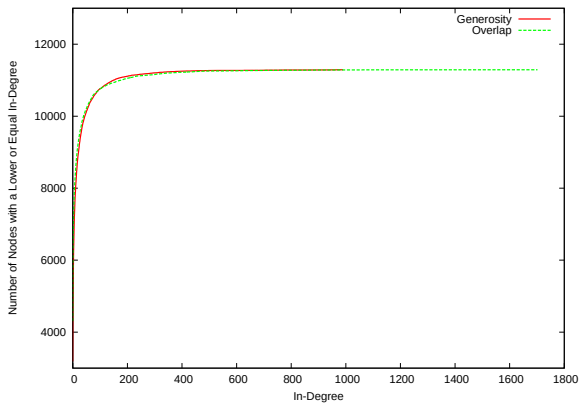


Figure 4: Peer Distribution for Severals Cycles in the Gossip-based Protocol

4.2.1 Peer distribution

Figure 4 represents the in-degree distribution of peers, *i.e.* the number of occurrences of the peer in other peers semantic list, depending on the gossip-based protocol cycle. At cycle one, the network is randomly initialised so that, the plot is a Gaussian-like, centered around 20, the view size chosen for the experiment. For all other cases, from the cycle 10, 90% of peers have got an in-degree that is lower than 10.



Strategy	Maximum In-degree
Overlap	1702
Generosity	990

Figure 5: Cumulative Distribution Function with Generosity Impact

4.2.2 Impact of peer generosity

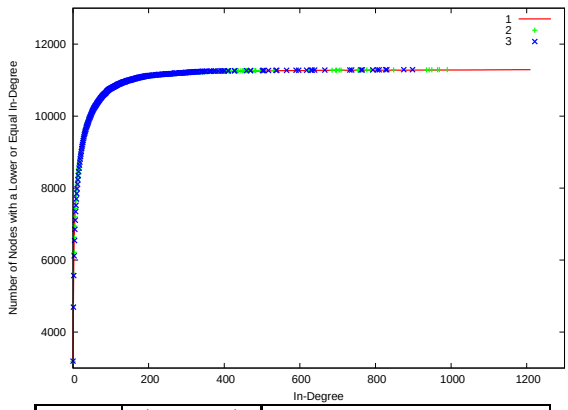
The goal of the proximity measure described in Equation 3 is to improve load balancing in the system. Figure 5 and 6 depict the cumulative distribution of in-degree. For each degree in the network is associated the number of peers having a in-degree lower or equal than the observed degree. For example, 11,112 peers have an in-degree that is lower or equal to 200.

Figure 5 shows the maximum load of most solicited peers in the system. The plot generosity has 990 as the maximum in-degree, whereas the overlap case, peer have a in degree greater than 1,700. Taking into account generosity helps to limit the maximum in-degree.

Figure 6 shows the influence of α and β parameters on the load balancing. The closer α and β, the better the load balance. The average hit rate is improved as well (14,53 for $\alpha = 1/2.1$ and $\beta = 1.1/2.1$ instead of 12,23 for $\alpha = 1/51$ and $\beta = 50/51$ at cycle 50).

4.2.3 Impact of file popularity

Figure 3 shows that the average hit ratio is greatly improved when popularity is taken into account (Equation 4) as opposed to a simple overlap strategy.



Plot	$(\alpha, beta)$	Maximum In-degree
1	$(\frac{1}{51}, \frac{50}{51})$	1210
2	$(\frac{1}{3}, \frac{2}{3})$	990
3	$(\frac{11}{21}, \frac{10}{21})$	898

Figure 6: Impact of α and β in the Distribution of Peers

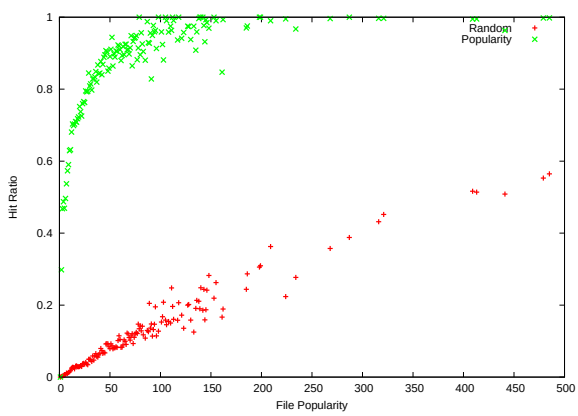


Figure 7: Hit Rate function of File Popularity

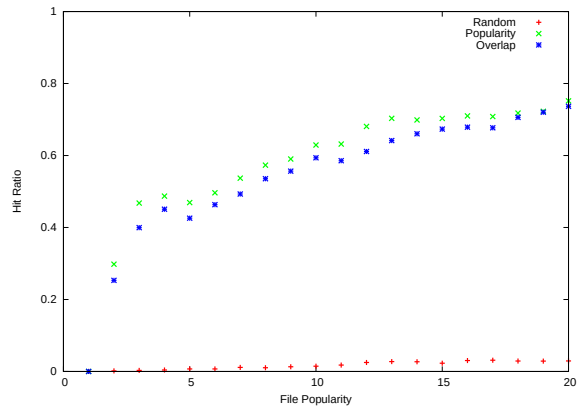


Figure 8: Hit Rate according to Unpopular Files

To evaluate file popularity, we measured the hit ratio for every file across all peers. To this end, at cycle 50, each peer sends a request for all of its files included in its content cache to all of its neighbours. Then, we calculated and associated the hit ratio for each file in the system. Figure 7 and Figure 8 report the average hit ratio according to file popularity.

Figure 7 shows the interest of using Equation 4 to measure semantic proximity, this increases significantly the hit ratio compared to a random approach.

Figure 8 zooms on Figure 7 to demonstrate that the less popular the files, the greater the impact. The difference between the different values for rare files is between 3% and 7%. The result for popular files is obviously the same for each case.

5 Tracking file popularity

Peer generosity can be easily computed locally to be used. Communicating peers only require to exchange their cache size. However, computing file popularity, measured as the number of replicas in the system, requires to scan the entire system. To determine the τ value of the proximity measure, each peer has to estimate the file popularity of each file of its content cache. The goal of the algorithm presented in this section is to compute file popularity in a fully decentralised way. To locally evaluate file popularity, we considered two approaches: (1) Random walks and (2) Gossip-based protocol.

5.1 Random Walk

In this approach, each peer periodically initiates a random walk for each file f through the system to determine its popularity. At each step, the request is forwarded to a random neighbour. In addition, each peer on the path sends a message back to the initiator specifying if it owns f. By aggregating the answers, an estimation of each file popularity can be computed locally: number of positive answer divide by the TTL (*Time-to-live*) value.

We made early experiments of this approach which has not proved efficient at all. We observe that below 1000 hops, the result is not significant (close to 0). This is partly due to the distribution of file as well. Using a 1000 hop random walk, the popularity was estimated correctly for 80% of the files. However, this approach has a huge overhead.

5.2 Gossip-based protocol

Alternatively a gossip-based protocol can also be used to collect some informations across the system. We use the same protocol structure as in Figure 1. At each cycle, the information exchanged between peers is related to the popularity value of all files they know about. When a peer joins the system, it sets this value to 1 for each file it owns. It sets this value to 0 for each discovered file in a gossip cycle. Then, each peer keeps the average of each popularity they exchange. For each file, we compute the following limit:

$$pop_f = \frac{\text{number of replicates in the network}}{\text{number of peers in the network}}$$

Experimental settings In order to evaluate this approach, the same discrete event-driven simulator and the same eDonkey workload is used. The evaluation compares the estimate value of popularity files and the value computed by the simulator based on a global knowledge of the system. Representative peers[6] are randomly chosen among the subset of peers.

Results The gossip-based protocol has a reasonable overhead: at each cycle, every peer sends one message containing a file ID and an associated popularity value to only one other peer. Figure 9 shows the results of the simulation. At cycle 6, 60%

[6] A representative peer have a large diversity cache content, including several popular and rare files

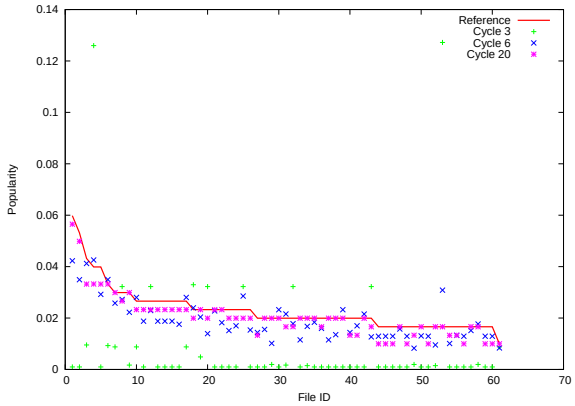


Figure 9: Detection with a Gossip-based Protocol

of the values are well ordered and after cycle 20, it represents 80% of the value well align.

The main overhead of this approach is the memory consumption: each peer needs to maintain information about each file it encounters. We evaluate the maximum memory load at 10 MBytes on each peer, for 1,000,000 files in the system.

6 Conclusion and future work

Semantic proximity has been identified as a relevant metric to improve search in peer-to-peer file sharing system. However, so far, very simple, and yet efficient approaches, have been proposed relying mostly on the recent history of requests. In this paper, we evaluated a finer-granularity semantic proximity measure to capture and exploit the semantic relationships observed between peers in a file sharing system. The goal of the proximity measure was to take into account both file popularity and peers generosity in the semantic measure as those factors had been previously identified as potential biases. We integrated the resulting measure in a gossip-based protocol, where links between peers were set according to their semantic proximity measure. We also proposed a fully decentralised algorithm to compute file popularity.

Based on simulation results, we observed that considering the peer generosity greatly improves the load balancing. Considering popularities of shared files can enhance average hit ratio and improve the localisation of rare files.

One perspective is to integrated this semantic proximity measure in other work which used a simple overlap metric as in [23]. Moreover, we evaluate this semantic measure in the context of a gossip-based protocol but it can be applied to other P2P networks as well.

References

[1] The eDonkey 2000 project. http://www.edonkey2000.com/.

[2] The Gnutella project. http://www.gnutella.com/.

[3] The KaZaA project. http://www.kazaa.com/.

[4] The Overnet project. http://www.overnet.com/.

[5] The Skype project. http://www.skype.com/.

[6] M. Castro, M. Costa, and A. Rowstron. Should we build Gnutella on a structured overlay? In *The 2nd Workshop on Hot Topics in Networks (HotNets-II)*, MIT, Cambridge, MA, USA, Nov. 2003.

[7] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: High-bandwidth multicast in cooperative environments. In *19th ACM Symposium on Operating Systems Principles (SOSP'03)*, The Sagamore, Bolton Landing, NY, USA, Oct. 2003.

[8] M. Castro, P. Drushel, Y. C. Hu, and A. Rowstron. Exploiting network proximity in peer-to-peer overlay network. Technical report, Microsoft Research, Cambridge, UK, 2003.

[9] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making gnutella-like p2p systems scalable. In *Special Interest Group on Data Communications (ACM SIGCOMM'03)*, San Diego, CA, USA, Aug. 2003.

[10] A. Crespo and H. Garcia-Molina. Semantic overlay networks for P2P systems. Technical report, Database Group, Stanford University, Stanford, CA, USA, Sept. 2002.

[11] P. Druschel and A. Rowstron. Past: A large-scale, persistent peer-to-peer storage utility. In *The 18th ACM Symposium on Operating Systems Principles (SOSP'01)*, Lake Louise, AL, Canada, Oct. 2001.

[12] S. Handurukande, A.-M. Kermarrec, F. L. Fessant, and L. Massouli. Clustering in peer-to-peer file sharing workloads. In *The 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04)*, San Diego, CA, USA, Feb. 2004.

[13] S. Handurukande, A.-M. Kermarrec, F. L. Fessant, and L. Massouli. Exploiting semantic clustering in the edonkey P2P network. In *The 11th ACM SIGOPS European Workshop (SIGOPS'04)*, Leuven, Belgium, Sept. 2004.

[14] S. Handurukande, A.-M. Kermarrec, F. L. Fessant, L. Massouli, and S. Patarin. Peer sharing behaviour in the eDonkey network, and implication for the design of server-less file sharing systems. Technical Report PI-1697, Institut de Recherche en Informatique et Systmes Alatoires, Rennes, France, Feb. 2005.

[15] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. The peer sampling service: Experimental evaluation of unstructured gossip-based implementations. In *ACM/IFIP/USENIX 5th International Middleware Conference (Middleware'04)*, Toronto, Ontario, Canada, Oct. 2004.

[16] B. T. Loo, R. Huebsch, I. Stoica, and J. M. Hellerstein. The case for a hybrid p2p search infrastructure. In *The 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04)*, San Diego, CA, USA, Feb. 2004.

[17] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *The 16th international conference on Supercomputing (ICS'02)*, June 2002.

[18] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *The IFIP/ACM International Conference on Distributed Systems Platforms* – Middleware *(IFIP'01)*, Heidelberg, Germany, Nov. 2001.

[19] S. Saroiu, K. Gummadi, R. Dunn, S. Gribble, and H. Levy. An analysis of internet content delivery systems. In *The 15th Symposium on Operating Systems Design and Implementation (OSDI'02)*, Boston, MA, USA, Dec. 2002.

[20] K. Sripanidkulchai, B. Maggs, and H. Zhang. Efficient content location using interest-based locality in peer-to-peer systems. In *The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM'03)*, San Francisco, CA, USA, Apr. 2003.

[21] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Special Interest Group on Data Communications (ACM SIGCOMM'01)*, San Diego, CA, USA, Aug. 2001.

[22] S. Voulgaris, A.-M. Kermarrec, L. Massouli, and M. van Steen. Exploiting semantic proximity in peer-to-peer content searching. In *The 10th International Workshop on Future Trends in Distributed Computing Systems (FTDCS'04)*, Suzhou, China, May 2004.

[23] S. Voulgaris and M. van Steen. Epidemic-style management of semantic overlays for content-based searching. In *The EuroPar'05*, Lisboa, Portugal, Sept. 2005.

A Appendix

Consider the following assumption:

- $\alpha + \beta = 1$
- $\alpha < \beta$

Furthermore, according to Figure 10

- $|\kappa_{B1}| < |\kappa_{B2}|$
- $|\kappa_{A1}| + |\kappa_{B1}| = |\kappa_{A2}| + |\kappa_{B2}| = T$
- $|\sigma_{A1,B1}| = |\sigma_{A2,B2}| = |\sigma_{A,B}|$

We want to evaluate if:

$$\xi_{A2}(B2) - \xi_{A1}(B1) \overset{?}{>} 0$$

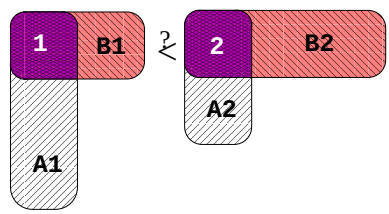


Figure 10: Order according to the same value of the overlap and different generosity considered from A

We have the following deductive reasoning:

$$
\begin{aligned}
&\xi_{A2}(B2) - \xi_{A1}(B1) \\
&= \left(\alpha \cdot \frac{|\sigma_{A2,B2}|}{|\kappa_{A2}|} + (1-\alpha) \cdot \frac{|\sigma_{A2,B2}|}{|\kappa_{B2}|}\right) \\
&- \left(\alpha \cdot \frac{|\sigma_{A1,B1}|}{|\kappa_{A1}|} + (1-\alpha) \cdot \frac{|\sigma_{A1,B1}|}{|\kappa_{B1}|}\right) \\
&= |\sigma_{A,B}| \cdot \alpha \cdot \left(\frac{1}{|\kappa_{A2}|} - \frac{1}{|\kappa_{A1}|}\right) \\
&+ |\sigma_{A,B}| \cdot (1-\alpha) \cdot \left(\frac{1}{|\kappa_{B2}|} - \frac{1}{|\kappa_{B1}|}\right) \\
&= |\sigma_{A,B}| \cdot \alpha \cdot \left(\frac{1}{|\kappa_{A2}|} - \frac{1}{|\kappa_{A1}|}\right) \\
&+ |\sigma_{A,B}| \cdot (1-\alpha) \cdot \left(\frac{1}{T-|\kappa_{A2}|} - \frac{1}{T-|\kappa_{A1}|}\right)
\end{aligned}
$$

Figure 11 shows the values of the last equation according to the different values of $|\kappa_{A1}|$ and $|\kappa_{A2}|$. We are able to affirm that for the most majority of cases, we have:

$$\xi_{A1}(B1) < \xi_{A2}(B2)$$

We extract the set of cases where $\xi_{A2}(B2) - \xi_{A1}(B1) < 0$. They are represented in Figure 12. We observe a side effect for larger values of κ_{A1} and κ_{A2}. When the two peers are highly generous, the order is reversed. However, this is not an issue in our approach since the order is important when cache sizes are very different and/or small.

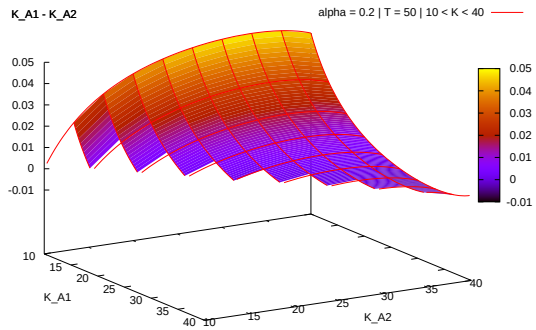


Figure 11: Evolution of $\xi_A(B)$ in function of κ_{A1} and κ_{A2}

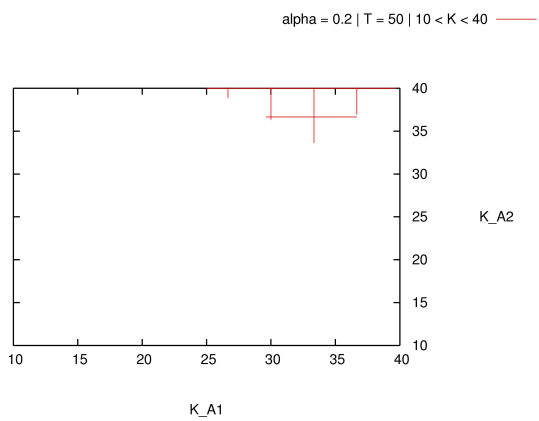


Figure 12: Evolution of $\xi_A(B)$ in function of κ_{A1} and κ_{A2}

Contents

1 Introduction and background **3**

2 System model **5**

3 Semantic proximity measure **6**
3.1 Cache overlap 6
3.2 Ignoring generous peers 7
3.3 Handling file popularity 8
3.4 Summary 9

4 Performance evaluation **9**
4.1 Experimental setting 9
4.2 Hit ratio results 11
4.2.1 Peer distribution 12
4.2.2 Impact of peer generosity 13
4.2.3 Impact of file popularity 13

5 Tracking file popularity **15**
5.1 Random Walk 16
5.2 Gossip-based protocol 16

6 Conclusion and future work **17**

A Appendix **20**