

Self-Clustering in peer-to-peer overlays

Anne-Marie Kermarrec

INRIA/IRISA

Campus universitaire de Beaulieu 35042 Rennes, France

E-mail: akermarr@irisa.fr

1 Introduction

Peer-to-peer overlay networks represents one of the most popular class of self-organizing systems. A peer-to-peer overlay connects peers in a scalable way and defines a logical network built on top of a networking infrastructure. They are able to dynamically and automatically adapt to frequent peer arrivals and departures, cope with high failure rate and link loss rate and are fully decentralized. Structured peer-to-peer overlays such as Chord [11] or Pastry [9], rely on a name structure and map object keys to overlay nodes. They can be used to implement efficiently distributed hash-table or application-level multicast for example. At the other end of the spectrum, in fully unstructured overlays such as pseudo-random graphs [4] or Gnutella-like networks [1], each peer is connected to arbitrary other peers. There are mostly used for flooding-based dissemination or file sharing applications. Semi-structured networks such as the KaZaA [5] or eDonkey [2] sit in between, usually relying on a hierarchical approach. Such overlays are mostly used in recent peer-to-peer file sharing applications. Among all potential applications a peer-to-peer overlay may serve, P2P file sharing applications have been deployed and widely used to the extent that they now represent a significant portion of the Internet traffic.

There is a recent trend to complement the basic overlay, irrespective of its structure, with additional connections to *friends* in the network [10, 12, 13, 8]. These additional *friendly* links can be used to increase the degree of trust or the availability [6],

the potential for recovery [13] or the efficiency for some applications where *friends* share some interest [10, 12]. Proximity routing where overlays links reflect the underlying network topology also falls into this category. The whole idea behind the *friendly* links is to cluster peers according to some criteria. One of the advantage of such an approach is that in most cases, the notion of friends is orthogonal to the structure of the underlying overlay.

However, in order not to endanger the scalability of peer-to-peer overlays, it is crucial to cluster nodes based on a very limited and local knowledge of the system on each peer. We call self-clustering this ability of each peer to be able to get connected to related peers. In this short paper, we discuss the issues of *semantic* self-clustering in the context of peer-to-peer file sharing applications in particular. We also briefly discuss the other potential situations that could benefit from self-clustering.

2 Emergent clustering in peer-to-peer file sharing workloads

Peer-to-peer file sharing workloads now represent the most significant part of the Internet traffic. Recent analysis has demonstrated that peer-to-peer file sharing workloads exhibit some clustering properties [3] reflecting the presence of semantic proximity between peers. Semantic proximity is related here to the degree of similarities in download patterns and cache contents. These relationship

may be exploited to improve the search mechanism based on the assumption that semantically related peers have a higher probability to be able to serve each other's requests.

Several proposals have been done along this line [10, 12] to detect semantically related peers and cluster them automatically. These additional clustering links complement the underlying overlay. Requests target first semantically-related neighbours before using the standard search algorithm as a back-up mechanism. Detecting semantic proximity without relying on any explicit classification is a challenging issue. Instead, recent download history and cache contents are used as implicit indicators of semantic proximity. The most natural candidate is the LRU strategy: each node maintains additional links to the x most recent uploader peers. LRU semantic links may be polluted (*i*) by popular items downloads [12]. (*ii*) by generous uploaders. Semantic clustering tend to be more frequent for non-popular files [3]. At the same time, requests to popular items are more frequent. Subsequently, peers might be linked to other peers serving requests to popular items without having any semantic relevance. In peer-to-peer file sharing workloads, 10% of peers provide approximatively 80% of the contents. Therefore, large cache contents tend to dilute the semantic relationship and at the same time generous uploaders tend to be very popular *false* semantic neighbours.

Despite the recent interest in extracting and using semantic relationship in peer-to-peer file sharing systems remains a difficult task. The adequate trade-off between the explicit information and the scalability still needs to be found.

3 Other candidates for self-clustering

In addition to the very popular peer-to-peer file sharing applications, a number of other applications may benefit, or already do, from clustering. Clustering is based on the detection of some form of affinity between peers in the context of a given con-

text. In this paragraph, we briefly go through other example of useful clustering.

Topologically-aware peer-to-peer overlay

Taking into account the underlying network topology in peer-to-peer overlays has been one of the first issue addressed by the research community. Peer-to-peer overlays, if oblivious to the physical network, may lead to a very high network traffic. In the context of the Pastry [9] or Tapestry overlays, peer candidates to fill the routing tables are selected according to a proximity metric. The ping metric has been traditionally adopted but other metrics, based for example on Internet coordinates could be chosen instead.

In the context of unstructured peer-to-peer overlay, we recently proposed a probabilistic algorithm, called the localiser [7], which refines the overlay so as to reflect the geographical topology. The localiser is fully decentralized and requires only some local geographical information, for example the communication cost between two peers expressed using the ping primitive, to perform a local rewiring between the two neighbours of a given peer.

In both cases, the clustering criterion has been the geographical distance. However, the same mechanisms could be applied to optimize the overlays according to another metric.

Grid computing Exploiting affinity between peers in grid computing may also significantly enhance the performance and is mostly related to data movement and placement. Grid applications usually manipulates very large data structures (such as matrices for example). Computation costs on such large data sets are very different whether the data is stored locally or remotely. Detecting affinity, between operations and data, is crucial for the performance and may impact both the initial data placement and the computations placement.

Exploiting affinity might also be relevant in the context of persistent data sets. Data persistence mostly relies on replication. In Grid computing, (*i*) moving large data set in not more expensive than moving small data set since nodes are connected

using a high-bandwidth dedicated network; and (ii) replication for persistence may be used for standard operations. Therefore, it might be particularly useful to detect affinity between nodes, based on the operations they perform and the data they store, so that data is preferably replicated on nodes having a high probability of using them in the future.

Interest-based clustering in publish-subscribe systems Another interesting application which could benefit from clustering is the content-based publish-subscribe applications for peer-to-peer systems: on one hand, peers subscribe to some well-defined patterns of events and on the other hand, publishers send events matching some of these patterns. In order to avoid a flooding of each event in the system followed by a filtering mechanism on each node, the publish-subscribe application is in charge of matching publications and relevant subscriptions somewhere in the peer-to-peer overlay without relying on any central filtering.

Self-clustering of peers sharing some interest represents a solution to the scalability issue. Special care has to be taken to cope with subscription patterns changes.

4 Conclusion

Peer-to-peer overlays have now become a major area of research in distributed computing. Their properties of self-organization and robustness make them very good candidate for many distributed applications. Basic overlays, whether they are structured, unstructured or semi-structured, can be complemented with additional links or optimized to reflect some clustering properties exhibited by the physical conditions or the applications. Self-clustering automatically and dynamically adapt the overlay as the peers behavior change over time without impacting its basic functionality. Despite the fact that some research effort has been recently devoted to this area, there are still major and highly relevant issues:

1. What are the relevant criteria for a given application?
2. How to capture the *-proximity in a automatic decentralized way relying only on a local knowledge of the system.
3. How to use this *-proximity to cluster dynamically and efficiently peers in a large-scale peer-to-peer system.

What makes the self-clustering property so interesting, in addition to its expected impact on the performance of peer-to-peer applications is its potential use in all kinds of peer-to-peer overlay networks.

Acknowledgments

I would like to thank Sidath Handurukande, Fabrice Le Fessant, Laurent Massoulié, Maarten van Steen and Spyros Voulgaris for many discussions on the use of semantic in P2P file sharing and Gabriel Antoniu and Luc Bougé for their insight on Grid applications.

References

- [1] The gnutella protocol specification v.0.4. <http://www.clip2.com/GnutellaProtocol04.pdf>, March 2001.
- [2] <http://www.edonkey2000.com>.
- [3] F. L. Fessant, S. Handurukande, A.-M. Kermarrec, and L. Massoulié. Clustering in peer-to-peer files sharing workloads. In *The Third International Workshop on Peer-to-peer Systems (IPTPS'04)*, 2004.
- [4] A. Ganesh, A.-M. Kermarrec, and L. Massoulié. Peer-to-peer membership management for gossip-based protocols. *IEEE Transactions on Computers*, 52(2), February 2003.
- [5] <http://www.kazaa.com>.

- [6] S. Marti, P. Ganesan, and H. Garcia-Molina. Dht routing using social links. In *The Third International Workshop on Peer-to-peer Systems (IPTPS'04)*, 2004.
- [7] L. Massoulié, A.-M. Kermarrec, and A. Ganesh. Network-awareness and a failure resilience in self-organising overlay networks. In *Symposium on reliable and distributed Systems (SRDS'03)*, Oct. 2003.
- [8] M. Naor and U. Wieder. Know my neighbor's neighbor: better routing for skip-graphs and small worlds. In *The Third International Workshop on Peer-to-peer Systems (IPTPS'04)*, 2004.
- [9] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *International Conference on Distributed Systems Platforms (Middleware)*, Nov. 2001.
- [10] K. Sripanidkulchai, B. Maggs, and H. Zhang. Efficient content location using interest-based locality in peer-to-peer systems. In *INFOCOM Conference*, 2003.
- [11] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM 2001*, San Diego, USA, Aug. 2001.
- [12] S. Voulgaris, A.-M. Kermarrec, L. Massoulié, and M. van Steen. Exploiting semantic proximity in peer-to-peer content searching. In *10th International Workshop on Future Trends in Distributed Computing Systems (FTDCS 2004)*, 2004.
- [13] H. Wang, Y.-C. Hu, C. Yuan, Z. Zhang, and Y.-M. Wang. Friends troubleshooting networks: towards privacy-preserving, automatic, troubleshooting. In *The Third International Workshop on Peer-to-peer Systems (IPTPS'04)*, 2004.