Building Dependable Peer-to-Peer systems

Koen Vanthournout Geert Deconinck
Ronnie Belmans
K.U.Leuven, Dept. Elektrotechniek, Kasteelpark Arenberg 10,
B-3001 Leuven-Heverlee, Belgium
Tel: +32/16/32.18.12 Fax: +32/16/32.19.85
Koen.Vanthournout@esat.kuleuven.ac.be

Abstract

Self-organizing genuinely distributed overlay networks (Peer-to-Peer networks) are expected to survive in the advent of failures. As such, they require a high resilience against node failures, message failures and network partitioning. This paper proposes three strategies to realize this: the use of a small-world topology, the use of the self-organization mechanisms for failure detection and failure handling and the use of cross-partition pointers to deal with network partitions. Simulations of a Peer-to-Peer resource discovery network that deploys these strategies confirm their validity.

1 Introduction

This paper focuses on some mechanisms and design principles to enlarge the resilience of Peer-to-Peer(P2P) networks against communication and node failures. P2P refers here to genuinely distributed overlay networks, i.e., overlay networks without coordinating units, in which all nodes take decisions based on local data only. This excludes SETI@home [9] and Napster [6], but includes Gnutella [4], Freenet [3], CAN [7], Tapestry [13], Chord [10] and Pastry [8].

Every P2P system consists of two main components: an overlay network and the application that uses it. The former allows the transmission of messages, not to a network address, but to a node with some semantic property: queries instead of messages. This semantic property can be expressed in text, keys or XML. Examples of applications that build on such an overlay network are file sharing [4, 3], key-value resolving [13, 7, 10, 8] and resource discovery [5, 11]. The overlay network strives to self-organize into a structure that supports and optimizes this query resolving. Multiple query forwarding strategies are possible and, dependent on that strategy, an optimal network structure can be envisioned. Optimal means here that every node connects to a selection of neighbors that match the network's construction criteria best (optimal node position).

It is possible to develop general solutions for these overlay networks, independent of the applications. This paper focusses on mechanisms to enlarge the dependability of unidirectional[1] overlay networks, i.e., their ability to survive and adapt to the loss of nodes and communication. The effects of those failures on the query forwarding or the applications in general, is not discussed.

The next section discusses the types of failures that are addressed here, followed by design principles and techniques to handle those failures. Experimental results, obtained from the simulations of a resource discovery P2P network (see Section 4), validate these mechanisms in Section 5.

2 Node and Communication Failures

Two types of failures are addressed here: failing nodes (fail-silent, crash semantics) and communications errors. Connection oriented style of communication with checksums and message confirmation/retransmission, is assumed (e.g., TCP/IP), which implies that the loss or corruption of single messages is hidden by the communication layer. Consequently, the communication faults the overlay network needs to deal with are temporarily unreachable nodes and network partitions.

A failure is assumed when no communication channel can be set up or if a connection is broken. Without additional information from the communication network and it is assumed there is none, it is impossible to further distin-

[1] Both overlay networks with unidirectional [4, 3] and bidirectional [13] links exist, but a detailed survey of their advantages and disadvantages would be beyond this paper's scope. The majority of the P2P networks deploys unidirectional links.

guish between a network failure and a node failure. Therefore, both types of failures must be treated identically.

Note the importance of the timeout of the communication layer for the failure detection, which is typically minutes. This should be set to a lower value to speed up failure-detection. A too small timeout can lead to false failure-reports, however. As such, the exact value depends on the application and its requirements.

Two specific overlay network problems are associated with the described failures:

- **Ghost nodes**: Ghost nodes are nodes that have failed or are unreachable but to which links still exist. This means that nodes that have not detected this failure yet, will attempt to route queries to those ghost nodes, which results in sub-optimal functioning. Even worse, nodes looking for new/better neighbors may receive pointers to ghost-nodes, which can result in the deterioration of the overlay network's structure and even new links to failed or unreachable nodes.

- **Overlay network partitioning**: When too many nodes fail at the same time (see Section 5) or when the communication network partitions, it is possible that the overlay network partitions: two or more sections emerge that have no pointers between them. This problem has been briefly addressed in [8], with infrequent random multicast node announcements as an (untested) solution. An alternative solution is addressed in Section 3. Note that temporarily unreachable or failed nodes present the same problem: unreachable nodes form a separate partition of size one and failed nodes become such a partition after their restoration. Thus, the solutions that solve network partitioning also cover isolated or restored nodes.

3 Dependability Mechanisms for P2P systems

What follows is a list of design principles and mechanisms to improve the dependability of P2P overlay networks:

- **Network topology**: Peer-to-peer networks find their likes in many networks in biology, technology and society, e.g., cellular networks, the World Wide Web and citation networks [1]. Many of these self-organizing networks exhibit behavior that cannot be explained by complete random or regular structures. Indeed, they exhibit properties most desirable for P2P networks: a small diameter, yet highly regular ('small-world' networks [12] and scale-free models [2]) and surprisingly tolerant for errors (as demonstrated in [1]). Especially the small world behavior is desirable for P2P networks: the high regularity allows efficient search strategies, while the low diameter keeps the number of hops per query low. And networks with these properties can tolerate large numbers of node failures without significant influence on that regularity and small diameter or before breaking down into several partitions. This is already successfully deployed by most key/value P2P networks: each node has a key and the nodes are organized into an overlay network that reflects the key space. A limited percentage of pointers to far nodes ensure the small diameter and small world behavior. An example is Chord [10] which deploys a circular key space or the P2P resource discovery network proposed in [11] (see Section 4). Summarized, it is a desirable property for P2P networks to exhibit a regular structure in which nodes point to neighbors in that structure, except for a small percentage of links that lead to a far-off node.

- **Self-organization**: Changes (new nodes, leaving nodes, changing functionality of the nodes, etc.) and failures represent a new situation to which the network must adapt. Consequently, mechanisms can be designed that cope with both. Since P2P networks typically face the changes described above, each node in the network has to reconverge periodically to its optimum position in the network, which is dependent on the current node composition. The same mechanisms can be used to recover from failed or unreachable nodes, posterior to their detection: a node adapts to the failure of the node as if it left the network, which allows for graceful degradation. The speed of this is defined by the period of the reconvergence cycle. The smaller the period the swifter the adaptation to changes, but at the cost of increased traffic.

- **Periodic description update**: If the P2P network is designed to cope with nodes with changing functionality, descriptions, contents, etc. (dynamic data), then each node must periodically contact its neighbors for update purposes. These periodic operations can also be used for node failure detection. Note that such a description update can be expensive if the description file size is large, which limits the minimum update period. If checksums are used instead of the complete file, the transmission of the file cannot only be limited to when it actually changed, but can also be done in parallel to the periodic update/failure detection messages.

- **Cross-partition pointers**: When partitioning occurs, above mentioned mechanisms will result in the formation of two or more internally optimized overlay networks. All links to the other partition will be permanently lost and the separated overlay networks will re-

main separated, even after the communication is restored, unless pointers to the other partition are manually inserted. A solution is to add a small FIFO buffer of fixed size to every node: a 'deceased' list. This list contains the n addresses of the n last nodes it used to link to, but that it detected to have failed. Nodes that announced their disconnection are excluded. These addresses serve as cross-partition references and every node should periodically attempt to contact the members of the deceased list. If this succeeds, then new links are formed to those recovered nodes and the partitions are merged or, alternatively, recovered nodes are reinserted in the P2P network, even if all network information in that previously failed node was lost. Note that this is a scalable mechanisms, since the total amount of memorized failed nodes grows proportionally with the number of nodes.

4 A Resource Discovery P2P Network

Simulations of a P2P network, conform the guidelines of Section 3, have been run as to verify those dependability mechanisms. The network in question is the resource discovery P2P network described in [11]. All resource providers announce their resources in an XML description file; resource users are also described by an XML file, but this then contains their interests. An overlay network is constructed based on a similarity metric that compares two XML files and yields a 'distance' value. The network will self-organize in such a manner that nodes are linked to the nodes that are as close as possible, which means that the distance value is as small as possible. The result is that the resources are clustered by type of resource and that the resource users are linked to the clusters that contain the resources for which they announced an interest, which allows the exploitation of group and time locality.

Next to a fixed number of links to neighbors, each node also has a 50% probability of creating a far link, which will lead to the node whose XML description file yields the greatest distance when compared to that node's description file. The result is a small world network: highly clustered, but with a small diameter, due to the far links [11].

Periodically, every node tries to find better neighbors (smaller distance value than for its current neighbors), as not only to adapt to new network compositions, but also to resource providers with modified resources or to resource requesters with varying interests. All nodes periodically update the copy of their neighbor's XML description file for the same purpose. The application-dependent period of the latter updates defines the response time of the network, although a small period comes at the cost of increased network load. Every communication, which is build on TCP/IP, serves for fault-detection: a broken connection or nonresponding node is assumed to have failed and this node is then added to the detecting node‘s local deceased list.

5 Simulations and Results

The simulations use units of dimensionless time. Because of this, the base unit for time used in the measurements is the period T by which nodes update the local copies of their neighbor‘s description files. The failure detection speed of the network is proportional to this update period, which makes it a good base for objective comparison. All tests are performed on a P2P network of 200 nodes with an average node degree of 8.8 (all nodes functional, after convergence). Failed nodes lose all internal memory of previous links, so when they recover, nodes cannot contact the network themselves (worst case situation): they become drifting nodes. Every running node tries to contact the members of its deceased list, which is of size three, with a period of $4T$.

Note that the time required to detect failures, to remerge partitions or re-insert restored nodes is sometimes merely a fraction of T. The reason for this is that the periodical description file updates of the different nodes are randomly spread in time. Combined with the average node degree of more than eight, this reduces the detection and remerging times to far below T and $4T$ respectively.

5.1 Random Node Failures

The first simulations look into the effects of the simultaneous failure of a portion of the nodes of the network. In a converged network of 200 nodes, random nodes are failed at the same instant. It is then measured how long it takes for the network to exclude these failed nodes, which become ghost nodes from the moment they fail until this exclusion. This is followed by the restoration of all failed nodes at time $1.5T$, which are now drifting unconnected, for they have no knowledge of any other node anymore. Figure 1 illustrates the simultaneous failure of 5% of the nodes. It requires $0.9T$ to exclude all ghost nodes and $1.95T$ to reinsert all recovered nodes into the overlay network. Figure 2 lists the results for a failure of 30% of the nodes. The ghost nodes are detected within $0.85T$, pulling all drifting nodes back into the overlay network takes $4.29T$, which is slightly more than the $4T$ period, used for the deceased list polling.

5.2 Attacks

Not random nodes are failed this time, but the nodes that have the most incoming links and that are consequently the most important nodes. Thus, the behavior of an efficiently conducted attack is simulated. The same measurements as in Section 5.1 are performed. The results for respectively

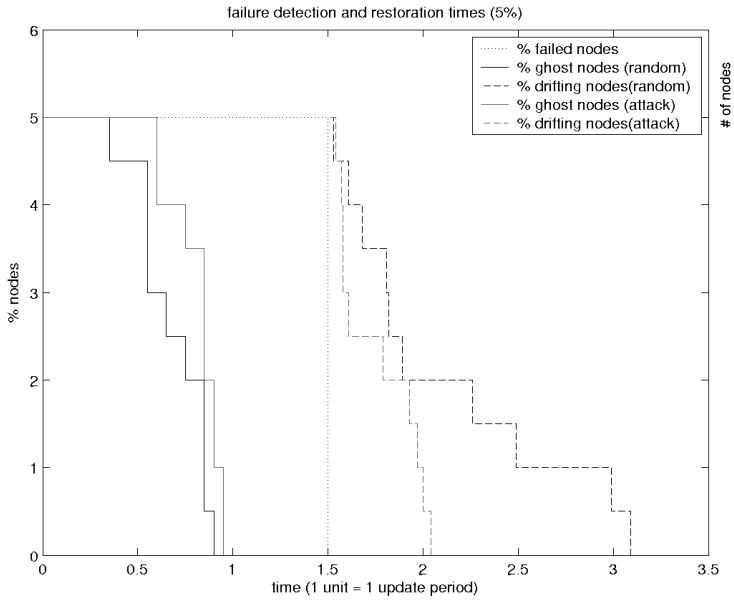

Figure 1. Effects of the failure of 5% of the nodes (random and attack selection): ghost nodes live length and time required to reinsert recovered nodes.

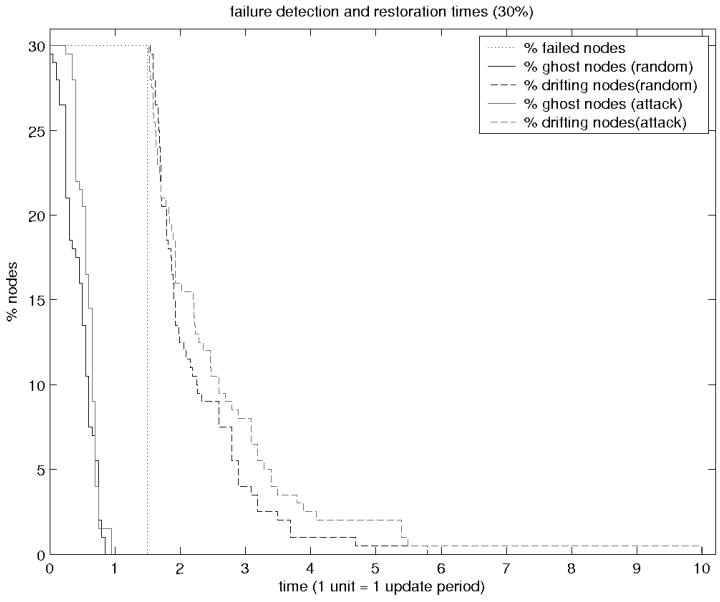

Figure 2. Effects of the failure of 30% of the nodes (random and attack selection): ghost nodes live length and time required to reinsert recovered nodes.

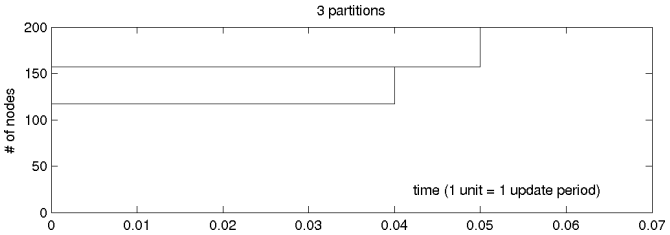

Figure 3. Size of the partitions after a partitioned network (3 partitions) is restored

5% and 30% of failed nodes can be found in the Figures 1 and 2. Because of the higher number of incoming links, ghost nodes have a longer live-time than in the case of random failures: $0.95T$ for both 5% and 30% of node failures. Remerging recovered nodes with the overlay network takes only $0.54T$ for the failure of the 5% nodes with the most incoming links (between 36 and 17). Indeed, these high numbers ensure a large number of entries in the deceased lists, which increases the chance of early detection. When a large number of nodes is attacked, this effect is countered by the lower average number of incoming links of the failed nodes. The deterioration of the overlay network structure because of the removal of the most important nodes adds to this. This is illustrated by the time of $8.49T$, required to recover all of the 30% failed nodes.

5.3 Network Partitions

Finally, the ability to withstand partitions is evaluated. Two tests have been performed: one in which the communication network was split in three and one with four partitions. The latter caused the breakdown of the overlay network in nine different segments. Following sequence was each time executed: first the communication network partitions, followed by the partitioning of the P2P overlay network. Measurements start at the point where the communication network restores. The results can be found in Figures 3 and 4. In the first case, the three partitions are rejoined after $0.05T$. The nine partitions present a greater challenge: all but one partition, composed of a single node, are remerged after $0.18T$. The last partition joins the remainder of the nodes after $2.24T$. Remember that nodes only poll the members of their deceased list every $4T$. Combined with the temporal spread of the updates of the different nodes and the average node degree of 8.8, this proves sufficient for a swift recovery from a large number of partitions.

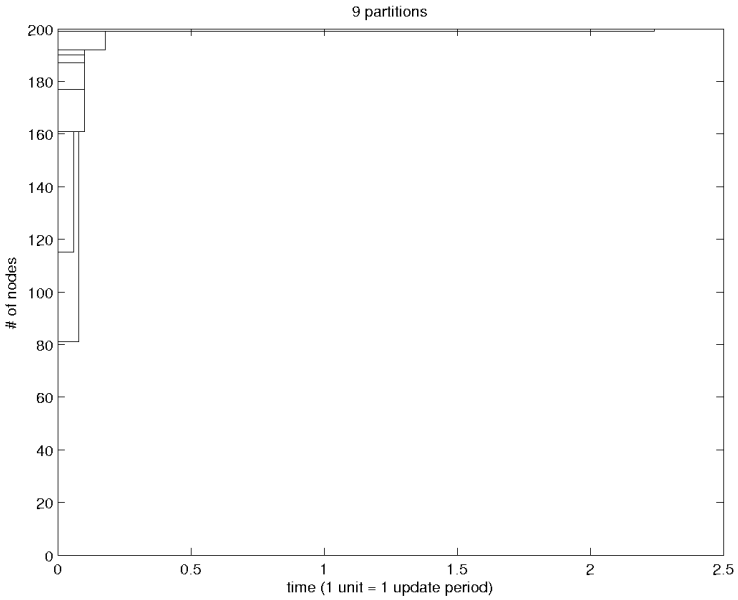

Figure 4. Size of the partitions after a partitioned network (9 partitions) is restored

6 Conclusions and Future Work

Three methods have been proposed to enhance the dependability of P2P networks: first of all, the deployment of small world topologies enhances the inherent ability of the network to withstand the loss of nodes with minimal influence on the regularity and diameter. Secondly, mechanisms that cope with dynamic networks and allow self-organization are also usable for failure detection and failure handling. And finally, each node can be enhanced with a list of cross-partition pointers to recover from network partitioning and to remerge recovered nodes with the network. Simulations of a P2P resource discovery network that used these techniques confirmed their validity. Further simulations are necessary, though, to investigate the effect of failures, not only on the P2P network‘s topology, but also on its functionality: the forwarding of queries. Additional dependability mechanisms may be needed to guarantee continued successful query forwarding in the advent of failures.

Acknowledgements

This work is partially supported by the K.U.Leuven Research Council (GOA/2001/04) and the Fund for Scientific Research - Flanders through FWO Krediet aan Navorsers 1.5.148.02.

References

[1] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–97, Jan 2002.

[2] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, Oct 1999.

[3] I. Clarke, O. Sandberg, B. Wiley, et al. Freenet: A distributed anonymous information storage and retrieval system. *Lecture Notes in Computer Science*, 2009:46–66, 2001.

[4] Gnutella. The gnutella protocol specification. http://rfc-gnutella.sourceforge.net.

[5] A. Iamnitchi and I. Foster. A peer-to-peer approach to resource location in grid environments. In J. Weglarz, J. Nabrzyski, J. Schopf, and M. Stroinski, editors, *Grid Resource Management*. Kluwer Publishing, 2003.

[6] Napster. www.napster.com.

[7] S. Ratnasamy, P. Francis, M. Handley, et al. A scalable content addressable network. In *Proceedings of ACM SIGCOMM*, pages 161–172, San Diego, USA, Aug 2001.

[8] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, Heidelberg, Germany, 2001.

[9] SETI@Home. http://setiathome.ssl.berkeley.edu/.

[10] I. Stoica, R. Morris, D. Liben-Nowell, et al. Chord: A scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, Feb 2003.

[11] K. Vanthournout, G. Deconinck, and R. Belmans. A small world overlay network for resource discovery. In *Euro-Par 2004*, Pisa, Italy, Aug-Sep 2004. Submitted for review.

[12] D. J. Watts and S. H. Strogatz. Collective dynamics of ’small-world’ networks. *Nature*, 393:440–442, 1998.

[13] B. Y. Zhao, L. Huang, J. Stribling, et al. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, 22(1):41–53, Jan 2004.