Fighting Erosion in Dynamic Large-Scale Overlay Networks

R. Baldoni S. Bonomi L. Querzoni A. Rippa S. Tucci Piergiovanni A. Virgillito
Dipartimento di Informatica e Sistemistica "A.Ruberti"
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italia
{baldoni,bonomi,querzoni,rippa,tucci,virgi}@dis.uniroma1.it

Abstract

Overlay management protocols have been introduced to guarantee overlay network connectivity in dynamic large-scale peer-to-peer systems. Some of these protocols have been specifically designed to avoid the partitioning of the overlay in large clusters (network breakage) despite massive node failures and the continuous arrivals/departures of nodes (churn). In this paper we identify a second effect connected to churn, namely network erosion. We show how erosion affects overlay network connectivity and point out that even a strongly connected overlay network, when exposed to continuous churn, can be disgregated. More specifically the consequences of erosion are shown, through an experimental study, in the context of overlay management protocols based on the view-exchange technique. We finally propose a connection recovery mechanism to be endowed at each node which is able to collaboratively detect node isolation and the presence of small clusters. This mechanism is shown to be effective in reducing the erosion of an overlay network exposed to continuous churn and to quickly recover its connectivity during stability periods.

Keywords: peer-to-peer, overlay networks, dynamic systems, churn evaluation.

1. Introduction

In the last decade the advent of peer-to-peer (p2p) computing introduced a new model of distributed computation where (i) the scale of the system can be very large, comprising up to millions of users (peers), (ii) each peer acts independently from all the others, actually precluding any form of centralized network-wide administration or management, (iii) each peer acts as a client of the service and cooperates with other peers to enable services for other participants, and (iv) the system, due to its size and the autonomy of each peer, is intrinsically dynamic as peers can join in or leave at any time.

In this context the basic problem that must be solved in order to build distributed applications is how to guarantee connectivity among participants. Connectivity is, in fact, the basic building block to enable network communications among peers. Modern p2p systems use, to this aim, an overlay network, i.e. a logical network connecting all the participants, whose maintenance is demanded to a specific protocol, namely an *Overlay Management Protocol* (OMP).

Motivation. When p2p systems grow up to very large scales, phenomena connected to the dynamic behaviour of nodes gain importance: the continuous arrival and departure of nodes, usually known as *churn*, can cause, if not properly addressed, *overlay network partitioning*.

OMPs for unstructured p2p systems, based on gossip approaches [4, 5, 2, 6] revealed to be very effective in the prevention of *major network breakages*, i.e. the partitioning of the overlay network in two (or more) clusters of approximately the same size. Overlay network breakages can be considered as catastrophic events that affect the system with a large and abrupt reduction of the overlay network connectivity. These protocols aim at building and maintaining, through some lightweight mechanisms, an overlay network with a random topology; the random topology is used to guarantee a low probability of major overlay network breakage even when a very large portion of nodes is abruptly removed.

However, overlay network partitioning can also take the form of a second distinct effect (beside network breakage): *network erosion*. Network erosion is a phenomenon, caused by churn, concerning progressive isolation of single nodes or tiny clusters that lose connectivity with the main cluster of the overlay network.

Contribution. Many OMPs underestimate this problem simply assuming that an isolated node , or nodes being part of tiny clusters, could eventually re-join the overlay [2] but without employing specific mechanisms. In this paper *we point out that fighting network erosion deserves the same*

attention as network breakages.

Erosion can be indeed so disruptive that a strongly connected overlay network exposed to continuous churn can be quickly disgregated. More specifically we show through an experimental study how badly network erosion affects view-exchange based overlay management protocols, like Cyclon [6] and ADH [2]. Overlay networks built through these protocols are, in fact, progressively eroded as long as the churn period lasts enough. Our study thus confirms that these OMPs were not designed to take erosion into account, and are thus not able to face its effects.

To fight erosion we propose a *connection recovery* mechanism, whose goal is twofold: (i) increase robustness of the overlay network during long periods characterized by churn and (ii) recover connectivity during periods of stability. To reach these goals we exploit a node re-join method that locally detects a status isolation from the network's main cluster. The presence of small clusters is also recognized and addressed by leveraging collaboration among nodes. Our experimental studies show how the connectivity recovery mechanism is able to reduce the effects of network erosion during long periods of time characterized by churn, improving the capacity of the OMPs to quickly react to topology changes; the experiments also show that, through this mechanism, the overlay network is able to quickly regain full connectivity when the system undergoes a stability period without churn.

Related work. The effects of network erosion have never emerged clearly in other works on OMPs for unstructured p2p systems [2, 6, 4, 5]. Voulgaris et al. in [6] and Ganesh et al. in [5] analyzed protocols' behaviour only in a static setting without churn to check the ability of these OMPs to resist to massive node failures. Allavena et al. in [2] and Eugster et al. in [4] addressed the overlay network partition problem considering prevention and recovery from large network breakages. Specifically, the former provides a completely decentralized solution while the latter assumes the presence of a set of fixed nodes in the system. Nevertheless, both solutions have not been experimentally evaluated, thus the effects of overlay network erosion did not came out. Finally, the first analysis of the behaviour of two different OMPs, namely SCAMP [5] and Cyclon [6], under continuous churn has been presented in [3]. In that work some problems related to continuous churn have been pointed out, but the paper did not provide any solution to address them.

The remainder of the paper is organized as follows. Section 2 introduces the reader to the details of two OMPs: Cyclon and ADH. Section 3 shows how overlays built and maintained through these two OMPs are affected by continuous churn. Section 4 introduces the *connection recovery mechanism*, while Section 5 shows its effectiveness through an experimental study. Finally, Section 6 concludes the paper.

2. View Exchange-Based Overlay Maintenance Protocols

An overlay network is a logical network built on top of a physical one (usually the Internet), by connecting a set of nodes through some links. A distributed algorithm running on nodes, known as the Overlay Maintenance Protocol (OMP), takes care of the overlay "healthiness" managing these logical links. The common characteristic of all OPMs is that each node maintains links to other nodes in the system. This set of links is limited in its size in order to favour system scalability and it is usually known as the *view* of the node. The construction and maintenance of the views should be such that the graph, obtained by interpreting links in views as arcs and nodes as vertexes, is connected, as this is a necessary condition to enable communication from each node to all the others. View maintenance can be realized through two main approaches: *structured* protocols use deterministic algorithms to update views' content, while *unstructured* ones usually rely on probabilistic solutions. OMPs employing the latter approach aims at building overlay topologies that closely resemble random graphs, and share with them nice properties like high connectivity and low network diameter. Thanks to these properties OMPs for unstructured p2p systems are usually considered as best candidates for highly dynamic settings like the ones we consider in this work.

OMPs for unstructured p2p systems differentiate among themselves with respect to the technique they employ to build and maintain views. They can be divided in two broad groups basing on the strategy used to manage node leaves:

Reactive Protocols - require each node to execute some algorithm before leaving the system [5].

Proactive Protocols - continuously adjust the network topology in order to allow nodes to leave without executing any specific algorithm [6, 2].

In this paper we focus the analysis of two proactive protocols, namely Cyclon [6] and ADH [2], due to the fact that there is nowadays common agreement in considering proactive protocols more suited than reactive ones to dynamic environments. This has been also confirmed by the simulation study presented in [3]. ADH and Cyclon are both based on a technique known as *view exchange* that requires nodes to continuously exchange part of their views in order to keep the overlay topology as close as possible to a random graph. Random graphs are characterized by strong connectivity, a property that is exploited to avoid network

partitioning: node departures or faults can, in fact, be simply ignored by the OMP as the random topology is supposed to remain connected despite node removals.

2.1. Cyclon

Cyclon [6] follows a proactive approach where nodes perform a continuous periodic view exchange activity with their neighbours in the overlay. The view exchange phase (named in this case "shuffle cycle") aims at randomly mixing views between neighbour nodes. Joins are managed in a reactive manner, through a join procedure, while voluntary departures of nodes are handled like failures (no leave algorithm is provided). A simple failure detection mechanism is provided in order to clean views from failed nodes.

Data Structures and Parameters - Each node maintains only a single view of nodes it can exchange data with. The size of the view is fixed and can be set arbitrarily. Each node in the view is associated to a local age, indicating the number of shuffle cycles during which the node was present in the view. A predefined parameter l defines the number of links exchanged during each view shuffle.

Join Algorithm - A node A joins the overlay network starting from a node (*bootstrap node*) among those already present in the network. The protocol starts then a set of independent random walks from the bootstrap node. The number of random walks is equal to the view size, while the number of steps per each random walk is a parameter of the algorithm. When a random walk terminates, the last visited node, say B, adds A to its view by replacing one node, say C, which is added to A's view using an empty slot.

Shuffle Algorithm - The shuffle algorithm is executed periodically at each node. A shuffle cycle is composed of three phases. In the first phase a node A, after increasing the age of all the nodes in its view, chooses its shuffle target, B, as the node with higher age among those in its view. Then, A sends to B a shuffle message containing $l-1$ nodes randomly chosen in A's view, plus A itself. In the second phase, when B receives the shuffle message from A, it replaces $l-1$ nodes in its view (chosen at random) with the l nodes received from A and send them back to A. In the final phase A replaces the nodes previously sent to B with those received from it. Overall, the result of one shuffle cycle is an exchange of l links between A and B. The link previously connecting A to B is also reversed after the shuffle.

Handling Concurrency - In the specifications given in [6], no action was defined in the scenario of two (or more) *concurrent* shuffle cycles, e.g. when a node A, during a shuffle cycle in progress with B, is selected as a target node by receiving a shuffle message from C. If concurrency is considered, the nodes sent by A to B can be modified by the concurrent shuffle involving A and C. To analyze the behaviour of Cyclon in concurrent scenarios we extended the original specification in order to address this situation: when nodes that should be replaced by A are no longer present in its cache, A replaces some nodes chosen at random.

2.2. ADH

ADH employs a slightly different strategy to maintain views. Each node periodically substitutes its whole view with a new one, which is built basing on information collected since the last view exchange. Even in this case joins are managed in a reactive manner, through a join procedure, while voluntary departures of nodes are handled like failures. Failure detection techniques are not used because crashed nodes are automatically discarded by the view exchange algorithm as time passes by.

Data Structures and Parameters - Like Cyclon, also ADH employs a single view for each node. The size of the view k is fixed and can be set arbitrarily. Two more parameters are considered: the *fanout* f and the *weight of reinforcement* w. Both are detailed later in this section.

Join Algorithm - Nodes joining the overlay network fill their initially empty views with the view of one of the nodes already in the system. ADH does not prescribe any specific method to choose this *bootstrap node*, as the OMP should be always able to balance the network approximating a random topology.

View Exchange Algorithm - Each node updates its view periodically, at the end of every *round*[1]. During a round each node collects:

- a list L_1 comprising the local views of f nodes chosen at random from its view;
- a list L_2 comprising those nodes that requested its view during the round.

At the end of each round these two lists are used to create the local view that will be used in the next round. The new view is built by choosing k nodes from both L_1 and L_2. The weight of reinforcement w ($w \in [0, \infty]$) is used to decide from which list a node must be picked: if $w = 0$ then all nodes are selected in L_1, if $w = 1$ nodes are selected with equal probability in L_1 and L_2, and, finally, if $w = \infty$ then all nodes are selected in L_2. This mechanism is used to keep the network "clean" of crashed nodes (that surely will not appear in L_2), while mixing views. For these reasons the authors of [2], with respect to the value of w, suggest "Larger is better and will be either 1 or ∞ on a typical implementation".

[1] In [2] the protocol is introduced in a synchronous environment where the notion of *round* is clearly defined. In an asynchronous setting, like the one we used to test the algorithm, the notion of round can be approximated with the time lapse between two consequent view exchange operations. In our setting rounds pertaining to different nodes are not synchronized.

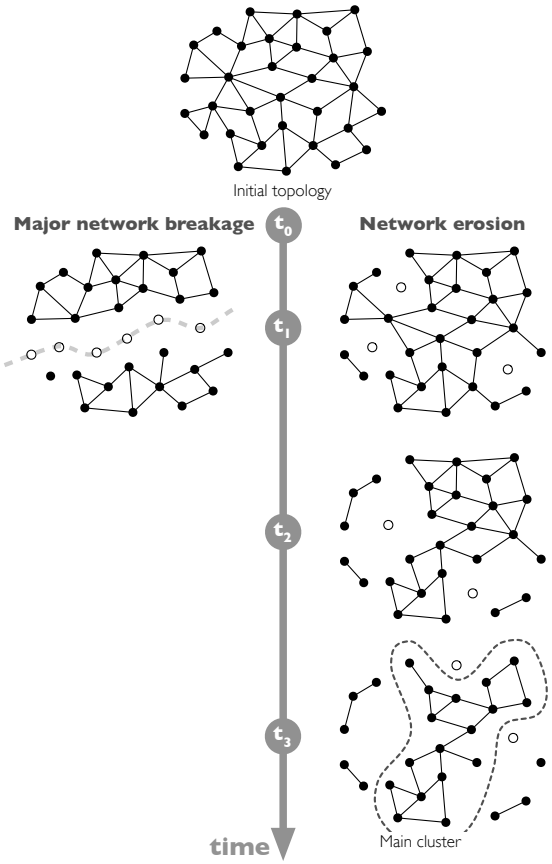


Figure 1. Breakage vs Erosion.

3. Overlay Robustness Under Continuous Churn

The protocols presented in Section 2 are able to build overlay networks whose topology approximates a random graph [6, 2]. Thanks to this property, systems built with these OMPs are supposed to be highly resilient to node removals. In a dynamic p2p scenario participants can enter or leave the system at their will, at any time. The global rate at which these actions occur is called the *churn rate*. Node removals happen continuously during some time periods, i.e. churn is not an instantaneous phenomenon but its effects are rather durable in time. During time periods characterized by sustained churn rates the time available to the OMP to repair the overlay network after a node departure, substituting dangling edges with valid ones, can become too short. Continuous churn can in these cases lead to loose overlay network connectivity an event that manifests itself in two ways: *major network breakages* and *network erosion*.

Network breakages are caused by the removal of one or more nodes which form the common frontier of two otherwise independent large clusters. When these nodes are removed no valid link exists that connects two clusters, thus nodes pertaining to distinct clusters cannot communicate. Figure 1 shows on its left side an example of a major network breakage. The net effect of a large network breakage is an abrupt and dramatic diminishment of the overlay network connectivity. OMPs based on gossip approaches like Cyclon and ADH revealed to be very effective in the prevention of such events. The random graphs they build and maintain, in fact, guarantee that, even when a very large number of nodes are removed, the probability of a major network breakage is extremely low. This result is clearly stated in [6], where it is actually tested only in static scenarios where nodes are removed all at once, and in [2].

Network erosion is a subtler phenomenon, endemic in systems affected by continuous churn, which progressively detaches single nodes or tiny clusters from the frontier of the main network cluster. This frontier is constituted by those nodes whose neighbors have been progressively removed, and whose views contain dangling edges. These nodes are indeed weakly connected to the main cluster and further neighbor removals can quickly bring them to a state of complete isolation. An example of progressive network erosion is depicted on the right side of Figure 1. It's important to note that erosion, contrarily to network breakages, is a phenomenon which affects progressively the overlay network, but whose effects are nevertheless dramatic. If not properly addressed erosion can, in fact, lead to the disgregation of the overlay network, quickly reducing a strongly connected cluster to a "dust" of isolated nodes.

In the next sections we will show through the results of an experimental study how much network erosion can affect connectivity in an overlay network built and managed by the two OMPs previously introduced, but before delving into the results, let us introduce the environment used to conduct our tests.

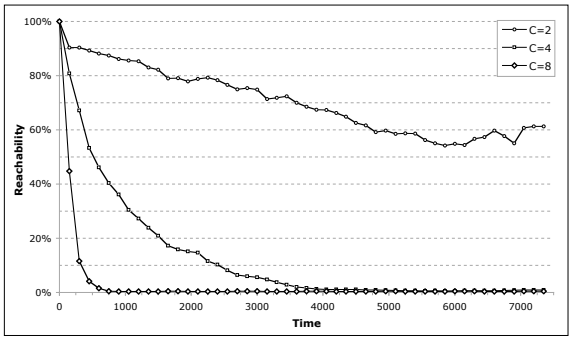
3.1 Simulation Settings and Measured Metrics

Our test were aimed at the evaluation of two metrics:

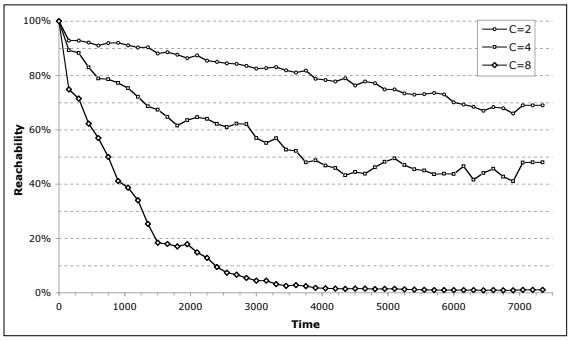
Definition 1. ***Churn Rate***
With Churn Rate *C we identify the global rate at which join and leave operations occur. In particular at each time unit, C new nodes invoke the join operation and C nodes in the overlay invoke the leave operation*

This rather simple churn model, brought from [2], was chosen to maintain the system at a constant size during tests.

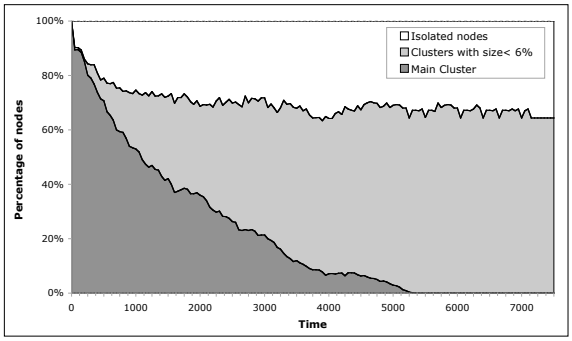
Definition 2. ***Reachability*** *With* Reachability *R we identify the average percentage of nodes that can be reached from any node in the overlay.*



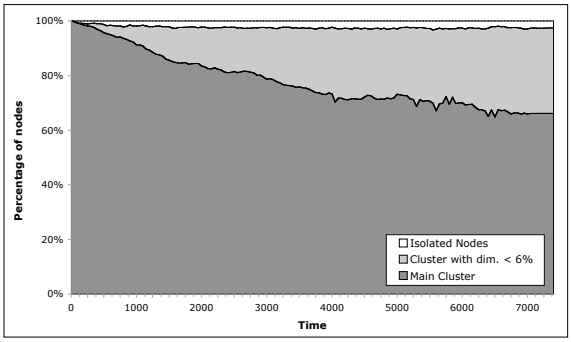
(a) Cyclon: evolution of R with different churn rates.



(b) ADH: evolution of R with different churn rates.



(c) Cyclon: evolution of overlay clustering with $C = 4$.



(d) ADH: evolution of overlay clustering with $C = 4$.

Figure 2. Evaluation of Cyclon and ADH under continuous churn.

It's important to note that R is strictly related to the connectivity of the overlay network, as any value lower than 100% indicates that at least one node cannot be reached by at least one other node.

To analyze the network erosion effect and how OMPs behave when a sustained churn rate is present, we implemented and tested in a simulated environment (provided by Peersim [1]) both Cyclon and ADH. A run of each protocol was divided in three distinct periods: *creation*, *churn* and *stability*. During the creation period nodes join the system until a predefined network size N is reached. Neither leaves nor overlapped join operations occur during this phase. During the churn period, nodes continuously join and leave the network at a given *churn rate* C. Leaving nodes are chosen uniformly at random from the network population. The churn period ends after 7000 time units. At the end of this period the number of nodes in the overlay network is still N, while the total number of nodes that joined/left the system depends on the specific churn rate C. N was set to 1000 in all experiments[2]. Message transmission delays vary uniformly at random between 1 and 10 time units. 10 independent runs were made for each experiment.

3.2 Evaluation of OMPs

Figures 2(a) and 2(b) report curves showing the evolutions of R's value as time passes by for Cyclon and ADH respectively. Different curves in the same graph represent different churn rates (C=2,4,8).

The curves show that reachability is strongly affected during the churn period. At the beginning the curves undergo a steep descending slope that is mainly due to the join of new nodes that are immediately considered as part of the system even if their views are initially empty; these nodes will affect negatively R until their join procedures

[2]Further experiments, not reported here, show that, with the considered churn model, the total number N of nodes in the system does not influence the final results as long as the ratio C/N is kept constant.

end filling their views. During the churn period network erosion continuously affects the overlay network isolating single nodes: this effect causes the continuous diminishment of R. It is important to note that both Cyclon and ADH show in these graphs the same behaviour even if with distinct absolute values.

At the end of the churn period the system regains a small percentage of reachability: this is mainly due to join procedures that end correctly in the first part of the stability period. The interesting point is that, nevertheless, both OMPs are not able to regain full connectivity after the churn period ends: after a short period of time, used by the OMP to "heal" what remains of the original network, the system stabilizes to a constant R value that is always below 100%. This problem can be imputed to isolated nodes, or to those residing in tiny clusters, whose messages are unable to reach many destinations.

Figures 2(c) and 2(d) confirm this result showing the evolution of node clustering for experiments conducted with $C = 4$. The curves represent percentage of nodes pertaining to the largest cluster (dark grey area), to clusters smaller than 6% of the whole network (light grey area), and isolated nodes (white area). These curves show that, even if ADH is more robust than Cyclon with respect to the effects of churn, the continuous arrival/departure of nodes negatively affects the main overlay cluster greatly reducing its size (with a consequent impact on reachability). From this point of view is fair to say that the analyzed OMPs are actually able to avoid large network breakages (we did not detect any massive network breakage during our tests), but node isolation, due to progressive network erosion, occurs very frequently.

4. Connection Recovery

As we showed in the previous section, node isolation occurs quite frequently when the overlay network experiences erosion due to continuous churn. The authors of [6] and [2] did not address explicitly this problem, but without any intervention isolated nodes will remain endlessly in this state.

In this section we introduce a *connection recovery* mechanism that can be added to both OMPs (and, generally speaking, to every OMP for unstructured overlay networks). Aim of this mechanism is to let nodes detect their isolation state and act consequently in order to regain connection to the main cluster of the overlay network. Moreover, our mechanism exploits cooperation among nodes to detect the presence of tiny clusters.

The basic idea of the connection recovery mechanism is simple: when a node detects that all the links in its partial view represent dangling edges, it triggers a new join procedure to regain connection to the system. In this way, isolated nodes will eventually re-join the system. Dead link detection is not done through some active mechanism but it is rather an indirect result of failed view exchanges (shuffles) with nodes that left the system.

To treat also nodes pertaining to small clusters (that will not satisfy the condition expressed above), we added a cooperative aspect to the basic mechanism.

The re-join procedure can be resumed in the following operations:

- n tries to re-join the system issuing a re-join request
- System assigns a bootstap node n_B to n
- n ping its bootstrap node n_B in order to know how many links n_B has in its partial view
- if $|partialviewn_b| < P$[3] then n reject n_B, ask for a new bootstrap node and puts n_b in a *low-connection list*.

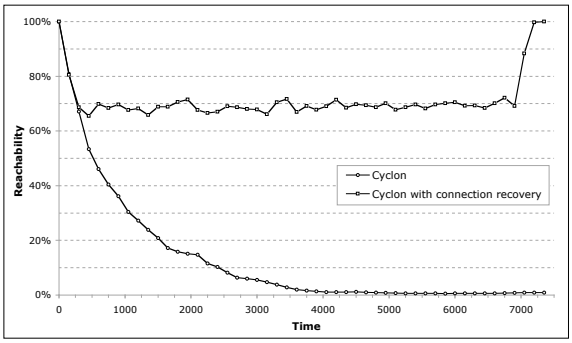
The re-join procedure is repeated until n encounters a node n^* which is able to guarantee a number of links larger than P. Once n has found a reconnection point, n warns all the nodes in its low-connection list that such a node exists in the overlay network. Note that n does not inform nodes in the list about the identity of n^*, otherwise a local star-like topology would be created around n^*.

Nevertheless, this sort of "signal" sent to some nodes will be interpreted by them as a clue that they are possibly part of either a small isolated cluster or a loosely connected part of the overlay: in consequence of this fact each node can independently decide to try a re-join even if its partial view is not empty. This decision is taken just looking at the current status of the view: if it contains a number of links still lower than P then the node will try to re-join the system. The parameter P actually influences the speed at which nodes re-join after detecting their isolation status. A possible solution to network partitioning based on connection recovery is also suggested in [2]. In this case the authors propose, to detect the presence of small clusters, a completely local approach where each node just check the variance of nodes in its view: if this variance is very low then the probability of the node being stuck in an isolated small cluster is high. This proposal was not evaluated in [2], is thus hard to compare its effectiveness versus our connection recovery mechanism.

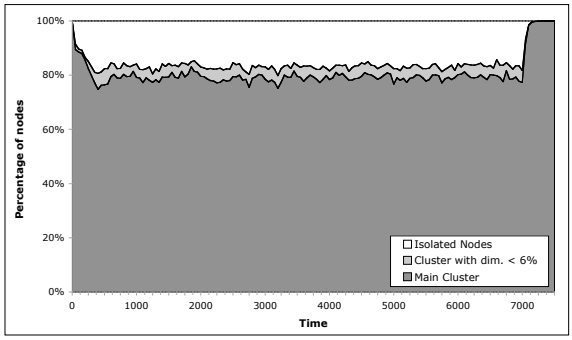
5. Evaluation

In this section we evaluate the connection recovery mechanism applied to the Cyclon OMP, in the same setting used for the previous test on $C = 4$.

[3] where P is re-join parameter



(a) Cyclon: impact of re-join mechanism on R.



(b) Cyclon with connection recovery: evolution of overlay clustering with $C = 4$.

Figure 3. Evaluation of Cyclon with connection recovery.

Figure 3(a) reports the value of R for a network employing our connection recovery mechanism; the same figure reports, for comparison purposes, the values obtained from experiments ran with the plain Cyclon OMP. The curve shows that, thanks to our mechanism, reachability R remains consistently higher with respect to the equivalent without re-joins. These curves point out a duplex effect caused by the connection recovery mechanism: a constant reachability value is maintained during churn periods and, as soon as a stability period starts, full connectivity is quickly regained, regardless of the R's values previously reached. It is worth noting that reachability values remain almost constant (or raise) as soon as the connection recovery mechanism starts to work at its full potential. The time needed for this to happen is proportional to both P and the view size: the mechanism will, in fact, take up to $viewsize$ shuffle intervals before starting the re-join for an isolated node.

The same results are confirmed by Figure 3(b) where the evolution of overlay clustering is shown. The curves highlight how the connection recovery mechanism quickly pushes isolated nodes and tiny clusters to rejoin the system, increasing the size of the main cluster. It is important to note that the amount of isolated nodes shown by this figure is heavily affected by newly joined nodes that are still waiting to complete their join procedures, and thus have empty views.

Given the self-healing capability brought in by the connection recovery mechanism, an important aspect that must be evaluated is the capability of the mechanism to converge to a stable state when churn is absent. In order to show this, we measured the evolution of the number of join operations induced by connection recovery.Tests were limited to 1500 time units as the algorithm did not show significant differences in its behaviour for longer tests in the same settings. As Figure 4 shows, the rate of join operations remains almost constant during the churn period, then immediately experiences a peak that is mostly due to the remaining isolated nodes that altogether try to re-join the system. As soon as the number of isolated nodes falls to zero the join rate drops. This actually means that the connection recovery mechanism remains active only for a limited time frame that is mainly linked to the length of the churn period, without causing further overhead when the system is stable.

Finally, we wanted to test if and how the addition of the connection recovery mechanism can alter the behaviour of the original OMP in terms of the type of network topology built. Our mechanism actually only trigger automatically node leave and join procedures, thus we expected no differences on the "quality" of the network built. This idea is confirmed by the curves shown in figure 5 where we report the in-degree distribution of nodes at the end of the simulation, for Cyclon with and without our connection recovery mechanism. The tests for this latter version were conducted in a scenario with $C = 4$. The curves clearly show that both implementations are able to build overlay networks with the same in-degree distribution, proving that connection recovery does not alter in any way the fundamental characteristics of the overlay.

6. Conclusion

This paper pointed out the importance of continuous churn as the first class enemy that must be fought in order to maintain an overlay network connected. This has been done

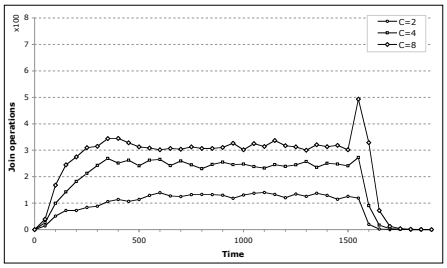


Figure 4. Cyclon with connection recovery: evolution of re-joins.

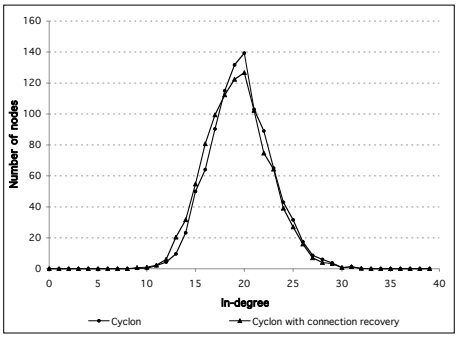


Figure 5. Cyclon: in-degree distribution at the end of the test with and without connection recovery.

by analyzing two gossip-based overlay management protocols based on view exchange, namely Cyclon and ADH. While these protocols are effective in avoiding large network breakages, under continuous churn we showed that they suffer network erosion, i.e., single nodes or tiny clusters that are progressively detached from the main component of the overlay network.

Through an experimental study we showed how disruptive network erosion can be, up to the point where an overlay network maintained by a specialized protocol can be quickly disgregated in a "dust" of isolated nodes or tiny clusters.

To address this problem we proposed a simple but effective connection recovery mechanism to be endowed at each node whose aim is to reduce the effect of network erosion under continuous churn and to completely recover overlay network connectivity as soon as churn ceases (even starting from a heavily disgregated network).

Even though these results are encouraging there are still various aspects that deserve further investigation. Our algorithm currently relies on a parameter P that is considered fixed and predefined: it would be interesting to devise approaches to let nodes independently evaluate at run-time the correct value for this parameter. Furthermore, the connection recovery mechanism presented in this work is based on collaboration among nodes: other approaches can be investigated to take into account p2p environments where nodes behave selfishly, and where malicious peers can try to voluntarily disrupt network connectivity.

References

[1] Peersim. http://peersim.sourceforge.net/.

[2] A. Allavena, A. Demers, and J. E. Hopcroft. Correctness of a Gossip Based Membership Protocol. In *Proceedings of the 24th ACM annual symposium on Principles of Distributed Computing (PODC05)*, pages 292–301, 2005.

[3] R. Baldoni, S. Bonomi, L. Querzoni, A. Rippa, S. T. Piergiovanni, and A. Virgillito. Evaluation of Unstructured Overlay Maintenance Protocols under Churn. Technical Report 2/06, MIDLAB, Dip. Informatica e Sistemistica, Università di Roma "La Sapienza", 2006.

[4] P. T. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec. Lightweight Probabilistic Broadcast. *ACM Transanctions on Computer Systems*, 21(4):341–374, 2003.

[5] A. Ganesh, A. Kermarrec, and L. Massoulie. Peer-to-Peer Membership Management for Gossip-based Protocols. *IEEE Transactions on Computers*, 52(2):139–149, 2003.

[6] S. Voulgaris, D. Gavidia, and M. van Steen. CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays. *Journal of Network and Systems Management*, 13(2), 2005.