Assessment of heuristics for self-stabilization in real-time interactive communication overlays

Pelayo Nuño

Department of Computing Science,
University of Oviedo,
Campus de Viesques, Gijón, 33204, Spain
Tel: +34 985 18 22 95 Fax: +34 985 18 19 86 E-mail: nunopelayo@uniovi.es

Juan C. Granda

Department of Computing Science,
University of Oviedo,
Campus de Viesques, Gijón, 33204, Spain
Tel: +34 985 18 26 38 E-mail: jcgranda@uniovi.es

Francisco J. Suárez

Department of Computing Science,
University of Oviedo,
Campus de Viesques, Gijón, 33204, Spain
Tel: +34 985 18 22 23 E-mail: fjsuarez@uniovi.es

Abstract:
Self-stabilization is an autonomic behavior closely related to self-healing. In multimedia communication overlays, self-stabilization copes with network disruptions by dynamically restoring data links and communication paths. The restoration of links may require reorganization of the overlay by establishing new connections among members or by modifying existing ones. Self-stabilization techniques are usually triggered asynchronously, either *on use* (during reorganization of the overlay) or *on event* (as a response to a failure), and several heuristics may be used when selecting link peers. This work assesses several heuristics to perform *on event* self-stabilization in multimedia communication overlays. A real-time communication overlay deployed following a full-mesh topology interconnecting a set of multicast islands is used as the assessment framework to evaluate the proposed heuristics. Intensive tests have been carried out to compare and assess heuristics with several overlay configurations and network conditions.

Keywords: self-stabilization; overlay networks; multimedia communications; autonomic computing

Reference to this paper should be made as follows: xxxx (xxxx) 'xxxx', *xxxx*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Pelayo Nuño is an associate professor in the Department of Computer Science and Engineering at the University of Oviedo. He received his M.S. and Ph.D. degrees in computer science from the University of Oviedo in 2009 and 2013, respectively. He has been involved in several projects within the Department of Computer Science and Engineering at the University of Oviedo. His research interests include multimedia networking and autonomic systems.

Juan C. Granda is an associate professor in the Department of Computer Science and Engineering at the University of Oviedo. He received his M.S. and Ph.D. degrees in computer science from the University of Oviedo in 2004 and 2008, respectively. In recent years, he has been working on several projects related to interactive multimedia services. His research interests include synchronous e-learning systems and multimedia networking.

Francisco J. Suárez is a professor in the Department of Computer Science and Engineering at the University of Oviedo and received his Ph.D. degree from that University in 1998. His research focuses on the evaluation of networked multimedia systems and he has been recently working on several projects related to interactive multimedia services.

1 Introduction

Self-healing is one of the fundamentals of autonomic computing (Kephart and Chess, 2003). This property refers to the ability of a system to identify, diagnose and recover from unforeseen interruptions and errors which might compromise its performance (Ganek and Corbi, 2003). Self-healing is an inherent property of multimedia communication overlays. These are composed of many entities generating, receiving or forwarding media streams. These entities may suffer failures, so self-healing techniques are necessary to maintain continuity and quality of service. Self-healing can be broken down into several autonomic behaviors such as self-diagnosis and self-stabilization (Nuño et al., 2013). Self-diagnosis identifies failures and localizes and addresses the causes of these failures (Haydarlou et al., 2006), while self-stabilization takes action to recover the system. Formally, self-stabilization is the automatic response of a system facing failures in order to achieve a steady state from any arbitrary configuration (Dolev, 2000).

Self-stabilization is closely related to self-organization (Berns and Ghosh, 2009), which is widely present in multimedia communication systems (Sterrit, 2005). In fact, self-stabilization always relies on self-organization in multimedia communication overlays to dynamically re-built data links or change the role of some of the members when dealing with member failures. Self-stabilization uses underlying self-organization techniques to restore communication paths between overlay members. The techniques implementing self-stabilization in communication overlays can be classified according to several criteria (Nuño et al., 2013). One of these criteria is triggering, which refers to the instant when recovering actions are performed. Triggering can be synchronous, that is, regularly triggered, or asynchronous.

Self-stabilization techniques are usually triggered asynchronously (Nuño et al., 2013) and can be of two types: those triggered after a change, *on event*, and those triggered *on use* (Alima et al., 2004). In the *on use* approach, self-stabilization is achieved by taking advantage of the overlay ability to self-organize, leading to a steady state when failing members are discarded during the self-organization procedure (Zhang et al., 2005; Pianese et al., 2007). Thus, there is no immediate response after a member failure. Each member is usually connected to many others, so a failure in one or many members of the overlay should not compromise performance when reorganizations are frequent. On the other hand, members initiate the self-organization procedure after detecting a failure in the *on event* approach. Most multimedia communication overlays follow this approach (Takahashi et al., 2013; Granda et al., 2015).

Real-time interactive communication overlays are highly latency-dependent. To overcome this constraint, they can be organized in topologies with high node degree. One alternative is to interconnect multicast islands where IP multicast is available. Another widely implemented alternative is to emulate IP multicast communications at end systems using an application layer multicast (ALM) approach for group communications (Gau et al., 2009). Those using IP multicast are more efficient (Hosseini et al., 2007) and may use special entities, called unicast/multicast reflectors. Reflectors are deployed in each IP multicast group, forwarding incoming data to several destinations, which are usually other reflectors or other overlay members. Minimum latency is achieved when reflectors are deployed following a full-mesh topology, since data between groups traverses a maximum of two reflectors. *On event* asynchronous triggering is the most suitable approach to implement self-stabilization in these overlays. Firstly, the reaction after a failure is immediate, so the disruption time is minimized. Secondly, since self-stabilization is triggered only when needed, it introduces less overhead than the *on use* approach as a lower number of control messages is required.

In this paper several heuristics to perform *on event* triggering self-stabilization in real-time interactive communication overlays are assessed. These heuristics are applied to a full-mesh overlay of reflectors supporting real-time interactive activities in corporate environments. In particular, the full-mesh overlay taken as a reference has been successfully deployed jointly with a synchronous e-training tool to develop several courses for training human resources (Granda et al., 2013). The heuristics are assessed under several overlay topologies and network conditions, analyzing the self-stabilization ability of the overlay network to redirect the overlay members to the remaining reflectors when a reflector fails. Although it is easy to deploy a reflector in each corporate site, traffic from real-time interactive activities, such as synchronous e-training sessions, must share the available network resources with other data which is usually crucial for the daily tasks of the corporation. This may lead to a reduction in the real available network resources, so efficient data delivery among the reflectors of the overlay is a must.

The remainder of the paper is organized as follows. Section 2 introduces the architecture of the overlay. Several heuristics to perform self-stabilization in such an overlay are presented in Section 3. These heuristics are tested in Section 4 and the results exposed in Section 5. Finally, Section 6 contains the concluding remarks.

2 Architecture of the Overlay

Global IP multicast is not widely deployed for technical and security reasons. However, IP multicast is feasible in each site of a corporate network. Under these circumstances, IP multicast can be used to deliver data within a corporate site, taking advantage of its optimal utilization of network resources. The real-time interactive communication overlay used as the assessment framework in this paper (hereinafter referred to as the *overlay*) is tailored to corporate networks and has been successfully used in e-training activities with video and audio streams, shared whiteboards and telepointers (Granda et al., 2013).

The overlay combines IP multicast and unicast delivery of multimedia data. Traffic is delivered using IP multicast among members within each site, while traffic among different sites is delivered using unicast. Therefore, the

overlay connects isolated IP multicast domains so members can participate in real-time interactive activities regardless of their locations in the corporate network. This is accomplished by deploying multiple reflectors, one in each corporate site. A reflector acts as a proxy between the IP multicast domain where it is located and the rest of the reflectors in the overlay.

The overlay is composed of three planes managed by a central entity called Rendezvous Point (RP). The reasons behind a centralized management are both simplicity and bandwidth saving, due to a lower number of control messages. The overall architecture of the overlay is depicted in Fig. 1.

The first plane represented in Fig. 1 is the reflector mesh (white dashed lines). The reflector mesh is responsible for forwarding multimedia data among reflectors (squares) using the Real-time Transport Protocol (RTP). A reflector is deployed in each corporate site, which can be considered as a multicast island, and all of them are interconnected composing a full-mesh topology. Overlay members (white circles) are distributed into several multicast islands, sending and receiving RTP streams to and from reflectors, using IP multicast.

Occasionally, when a reflector fails, members co-located in the same multicast island are redirected to other reflectors in order to maintain continuity of service. Redirected members use unicast instead of IP multicast to communicate with reflectors, as IP multicast is not supported between multicast islands. Therefore, a reflector sends and receives unicast data to and from redirected members and peer reflectors, but uses IP multicast delivery with members within its multicast island. The RP is responsible for re-directing members in the multicast island of a faulty reflector to another active reflector in the reflector mesh. The RP can use various heuristics to select which reflector is the best place to which to redirect each member.

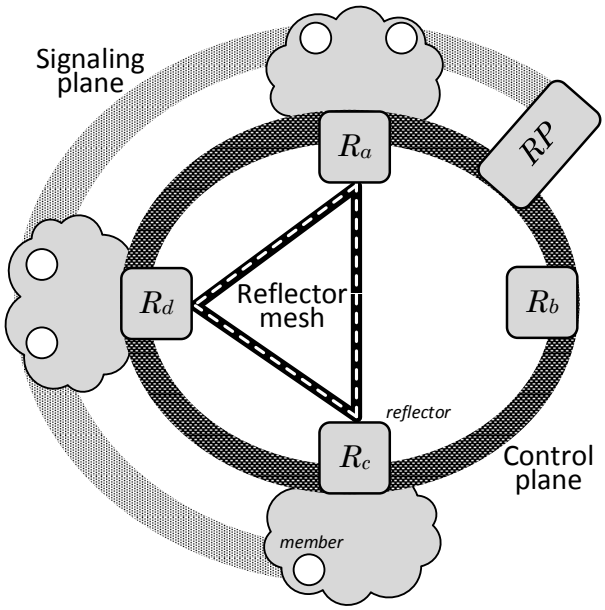


Figure 1: Architecture of the overlay

A control plane (dark shade) interconnects all the reflectors with the RP. The RP can change the reflector mesh organization including a new reflector or excluding an active reflector. A reflector is included in the mesh when the first overlay member from its multicast island joins. On the other hand, a reflector is excluded from the mesh when the last overlay member from its multicast island leaves. This process prevents the overlay from wasting network resources in the reflector mesh, as data streams are not forwarded to reflectors with no overlay members. The RP sends control messages to the reflectors to carry out these reorganizations. Similarly, the RP modifies the organization of the reflector mesh when a reflector fails. The RP notifies the remaining reflectors in the mesh that they must stop forwarding traffic to the faulty reflector.

Finally, the signaling plane (pale shade) allows members to join and leave the overlay. The Session Initiation Protocol (SIP) is used for session management. The RP acts as the SIP focus while the overlay members act as SIP clients.

3 *On Event* Self-Stabilization Heuristics

When a reflector fails, the overlay triggers the self-stabilization technique to redirect each member located in the multicast island of the faulty reflector to another reflector. The RP chooses the reflector that will support each redirected member, while penalizing the performance of the overlay as little as possible.

There are several works which analyze self-stabilization heuristics in distributed networks (Saia and Trehan, 2008) (Baresi et al., 2013). They are mainly focused on analyzing self-stabilization within the scope of highly variable overlays, where the overlay topology can be dynamically modified and high member churn is expected. In this paper, heuristics are applied to an established overlay supporting real-time interactive communications, which means that no further members or multicast islands are expected to be included in the overlay once the interactive session has started. In addition, the reflector mesh must be deployed in a full-mesh topology to achieve the minimum latency, as data traverses a maximum of two reflectors between overlay members. No topology changes are allowed, to avoid latency increases. This requirement is critical for real-time interactive activities. Therefore, the number of candidate reflectors taken into account by heuristics is fixed and limited.

Next, several heuristics that can be used to determine which reflector is the best candidate for self-stabilization are presented. Although they are described according to the overlay taken as assessment framework, they can be easily implemented in other centralized or distributed overlays.

a. *Random:* The RP chooses the reflector randomly.

b. *Number of members:* The RP chooses the reflector supporting the least number of members.

c. *Number of redirected members:* This heuristic is quite similar to the previous one, but the RP chooses the

reflector supporting the least number of members through unicast connections. In other words, this heuristic does not consider the multicast members supported by each reflector.

d. *Number of sources:* The RP chooses the reflector supporting the least number of members acting as a data source. Note that heuristics from *b* to *d* only consider characteristics of the members such as the distribution among multicast islands, or their roles in the session.

e. *Balance of the workload among reflectors:* Workload is measured in terms of the number of data streams that a reflector receives and forwards using unicast connections. Multicast streams are not considered since they represent a constant workload for all the reflectors.

Let θ_i be the number of unicast data streams sent and received by reflector i. This can be computed using the number of reflectors (S), the number of members (n), and the number of multicast and unicast members supported by the reflector i (m_i and u_i respectively) when each overlay member generates one data stream:

$$\theta_i = (S + n - 2)u_i + (S - 2)m_i + n \quad (1)$$

Firstly, the streams of all the members are forwarded to unicast members and the streams of unicast members are forwarded between reflectors, $(n-1)u_i + (S-1)u_i$. Secondly, the streams of the multicast members of the island of the reflector are forwarded to the rest of the reflectors, $(S-1)m_i$. Finally, the reflector receives all the streams of the members outside its multicast island, $n - m_i$.

Furthermore, a balancing parameter ρ_i can be defined in the range $(0, 1]$ to estimate the capacity of reflector i to forward data according to its initial available resources. A value of ρ close to 0 means a small amount of resources.

In addition, a metric $\delta_i = \theta_i/\rho_i$ is defined for a reflector, so the standard deviation of δ is used to calculate the workload distribution over the reflector mesh. Thus, this heuristic turns self-stabilization into a search problem where $\sigma(\delta)$ must be minimized, so a member is redirected to the reflector which minimizes:

$$\sigma(\delta) = \sqrt{\frac{1}{S-1}\sum_{i=1}^{S}\left(\delta_i - \overline{\delta}\right)^2} \quad (2)$$

f. *Proximity:* The RP chooses the reflector closest to the faulty reflector. The distance is calculated using the round trip time (RTT), which measures the propagation delay between neighboring reflectors (Draves et al., 2004).

g. *Ideal:* In an ideal situation, the RP chooses the reflector with the highest amount of available resources. This heuristic is usually quite difficult or even impossible to use, either because it requires extra control messages to estimate the available resources (and hence reducing them), or because network conditions change rapidly.

Table 1 Simulation settings

Reflector links average bandwidth (Mbps)	15
Reflector links standard deviation (Mbps)	5
RP link bandwidth (Mbps)	10
Max. number of concurrent audio streams	4
Audio stream bitrate (kbps)	15.2
Number of repetitions	5000

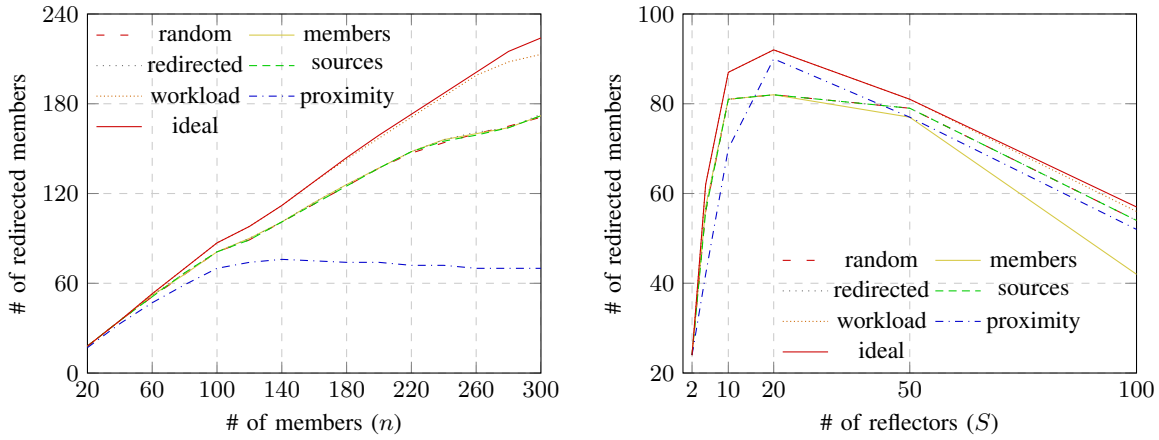
Finally, it should be noted that a second heuristic can be used as runoff criterion when the main heuristic selects several reflectors as best candidates. The *Random* heuristic is typically used in this case. In addition, network parameters used in heuristic *e* can either be established statically, or they can be dynamically computed as network conditions change, which would require the RP to gather additional information from reflectors. For simplicity, these parameters are established statically during the tests carried out. Analyzing the best approach to implement dynamic data gathering is beyond the scope of this paper.

4 Experimentation

The performance of heuristics can be assessed according to various metrics at entity level (computation time, memory usage, etc.) and network level. At network level the performance of heuristics can be determined by measuring the efficiency of the overlay after the stabilization process. Several metrics can be used such as communication delay between members, interarrival jitter, bandwidth consumption in the overlay or link stress (Di et al., 2010).

The metric used in this work is the available bandwidth in the links of the reflectors to the rest of the overlay. This is calculated after the stabilization process caused by a faulty reflector is completed. An efficient use of these links is critical, since they connect the corporate sites where the reflectors are located to the rest of the corporation. One heuristic is better than another when the reflectors in the organization of the overlay after the stabilization process require less bandwidth in these links.

Simulation tests are used to determine the maximum number of concurrent reflector failures supported by each heuristic, and hence the maximum number of members that can be redirected before the overlay reaches saturation or only one reflector remains active. Saturation occurs when one or more relays exhaust their available bandwidth. Subsequent random reflector failures are simulated in the overlay during an audio conferencing session. The members supported by the faulty reflector are redirected according to each heuristic until reaching overlay saturation. After each member is redirected, the number of members acting as data sources is randomly modified. The parameters used during the tests are summarized in Table 1.



(a) Overlay with 10 reflectors

(b) Overlay with 100 members

Figure 2: Maximum number of redirected members after subsequent reflector failures when varying (a) the number of overlay members, denoted by n, and (b) the number of reflectors, denoted by S

Although the overlay supports several multimedia types concurrently, a highly interactive audioconference has been simulated in the tests so each redirected member requires a low amount of extra resources (each member sends one or no audio streams). Thus, the heuristics can be clearly compared. If each member sent more data streams (audio, video, etc.) the comparison of the heuristics would be extremely difficult, since the resources required for redirected members grow exponentially due to packet replication (note the $n \times u_i$ factor in equation 1), and hence the reflectors would exhaust their resources quickly. In this case, the performance of all the heuristics would be almost identical.

The Binary Floor Control Protocol (BFCP) is used by the members of the overlay to request and release floors from the RP. Thus, several simultaneous audio streams can flow through the overlay (up to a maximum of four). Specifically, each member acting as a data source sends an iLBC audio stream to its reflector with a packet size of 20 ms resulting in a bitrate of 15.2 kbps. The overhead generated by different protocol headers, in this case RTP/UDP/IP, results in a network bandwidth consumption of 31.2 kbps per audio stream.

Finally, in order to test multiple configurations of the underlying network, the available bandwidth of each reflector is randomly changed between repetitions.

5 Results

Figure 2 shows the performance of each heuristic in terms of the number of redirected members. Figure 2a depicts this as a function of the number of members supported by an overlay with 10 reflectors, while Fig. 2b depicts the performance as a function of the number of reflectors for an overlay with 100 members. Both graphs assume that all the reflectors serve the same number of members supported per reflector at the beginning of the tests.

As can be seen in Fig. 2a, the heuristic based on the workload performs better than the others as the number of members increases. In fact, its performance is quite close to the ideal. The difference with the others heuristics is more noticeable in the scenarios with the highest number of members. As shown, in such scenarios the heuristic based on the workload starts diverging from the ideal. Proximity is the heuristic showing the lowest performance. The performance of the proximity heuristic decreases in scenarios with more than 140 initial members while the performances of the others increase. In such scenarios with a mid-high number of members, the proximity heuristic severely penalizes the reflector chosen to support redirected members. The heuristic selects the nearest reflector, which is the same for each member of the multicast island, so the higher the number of members per reflector, the higher the increase in bandwidth consumption for the reflector supporting redirected members.

In Fig. 2b the performance of the heuristics when varying the number of reflectors in the overlay is shown. Again, the performance of the heuristic based on the workload is quite close to the ideal and better than the others, but the difference is slight. The overall overlay performance increases with the number of reflectors until 20 reflectors in the mesh and decreases for higher numbers of reflectors. When increasing the number of reflectors from 2 to 20 the effort of forwarding traffic of redirected members among the reflector mesh is shared between more reflectors, so the performance increases. However, for a mesh with more than 20 reflectors, packet replication surpasses the benefits of sharing members between more reflectors. The higher the number of reflectors, the more bandwidth consumption on a reflector that is supporting a data source, since it has to forward traffic to more reflectors. The performance of the proximity heuristic is low for a small number of reflectors as the number of members supported by each one is high. As the number of reflectors increases, the proximity heuristic improves its performance and becomes a good choice for overlays with a middle number of reflectors. Finally, heuristics considering the distribution of members or their roles (members, redirected and sources) perform quite similarly. However, the heuristic based on the number of

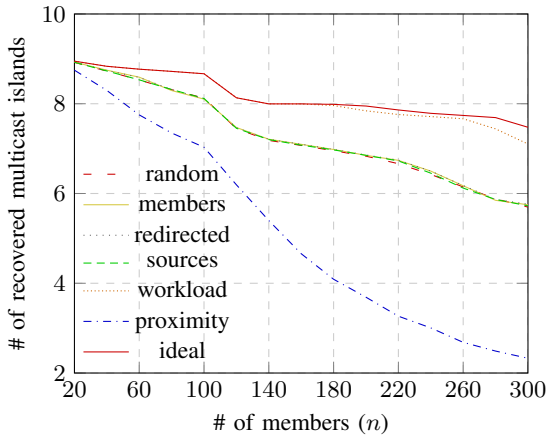
members is considerably worse for scenarios with a high number of reflectors. The reason behind this behavior is that the impact of redirected members and data sources in terms of bandwidth consumption is significantly greater than the rest of the members. Both redirected and sources heuristics take this into account when redirecting participants to reflectors, while the members heuristics does not differentiate members.

Figure 3 represents the same scenario as the previous figure but the number of fully recovered multicast islands is measured. A multicast island is considered recovered when all the members are redirected after its reflector fails. As can be seen in Fig. 3a, when the initial number of members per multicast island increases, the performance of the heuristics is lower since there is more traffic in the overlay. The heuristic based on the workload is the best choice, while the others have approximately the same performance. The exception is the performance of the proximity heuristic, which severely degrades while the rest of the heuristics suffer only slight degradation. On the other hand, as can be seen in Fig. 3b, almost all the heuristics have good performance, close to the ideal heuristic, when the number of initial reflectors is increased. The exception is the members heuristic, which has the worst performance. It is unable to recover half of the reflectors in the overlay for the worst scenario (100 reflectors).

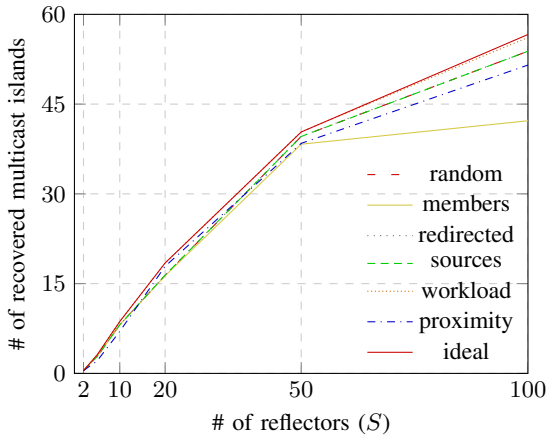
Figure 4 shows the number of members that can be redirected under different network conditions. Figure 4a depicts this as a function of the average bandwidth in the reflectors, assuming a standard deviation of 4 Mbps. Figure 4b depicts the number of redirected members as a function of the standard deviation of the available bandwidth in the reflectors, assuming an average bandwidth of 15 Mbps. Both scenarios consider an overlay composed of 10 reflectors and 100 members, with each reflector serving the same number of members. Furthermore, the minimum bandwidth assigned to a reflector during these tests is 1 Mbps.

As can be seen in Fig. 4a, the performance of the heuristics tends to converge as the average bandwidth increases. However, in scenarios with scarce bandwidth, the workload heuristic obtains better performance. Similarly, in these scenarios, both the redirected and members heuristics achieve slightly better performance than the sources heuristic. Fig. 4a also shows that the workload heuristic is an accurate approximation of the ideal.

Fig. 4b shows the performance of the heuristics when redirecting members according to the variation of the standard deviation of the available bandwidth of the reflectors. It shows that all the heuristics except the proximity heuristic work quite similarly when the standard deviation is small. For higher standard deviation (over 6 Mbps), the performance of the heuristics considering the distribution of members or their roles is noticeably worse than the workload and the ideal heuristics. In addition, for values of the standard deviation higher than 9 Mbps all the heuristics attenuate their degradation. This is because these values of standard deviation imply reflectors with massive available bandwidth (up to 50 Mbps), so if one of these reflectors remains active throughout the test, it will be able to support a large number of redirected members. This situation does not occur in



(a) Overlay with 10 reflectors



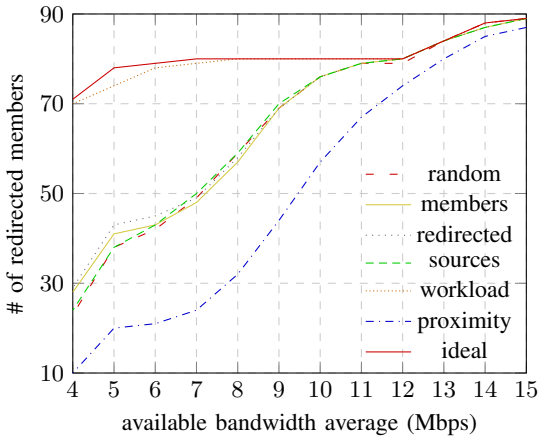
(b) Overlay with 100 members

Figure 3: Maximum number of multicast islands that can be recovered after subsequent reflector failures when varying (a) the number of overlay members, denoted by n, and (b) the number of reflectors, denoted by S

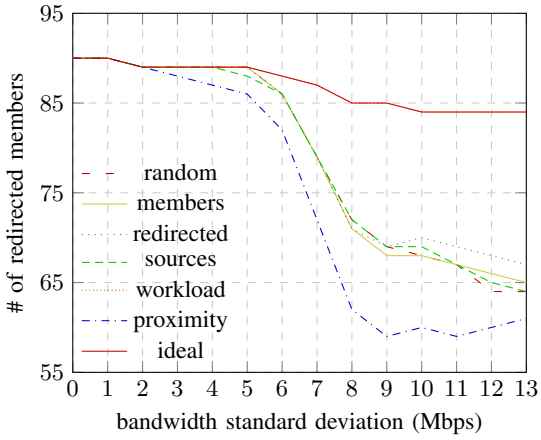
scenarios using mid-range values of standard deviation (from 4 to 6 Mbps).

Finally, Fig. 5 depicts the performance of the heuristics when the initial dispersion of members among reflectors is increased. Figure 5a shows the number of redirected members while Fig. 5b shows the number of fully recovered multicast islands. These tests are carried out considering an overlay composed of 10 reflectors and 100 members, so the average number of members per multicast group is 10. The standard deviation of members is varied in the tests keeping this average constant. In addition, an average bandwidth in the reflectors of 15 Mbps with a standard deviation of 5 Mbps is assumed during the tests.

As can be seen in Fig. 5a, the number of redirections that can be performed grows with the initial dispersion of members across the overlay. This occurs since there are scenarios where a reflector serving the highest number of members fails, slightly increasing the number of redirections during the test as compared to a balanced scenario. Again, all the heuristics except proximity perform quite well. However, the performance of the members heuristic compared to the

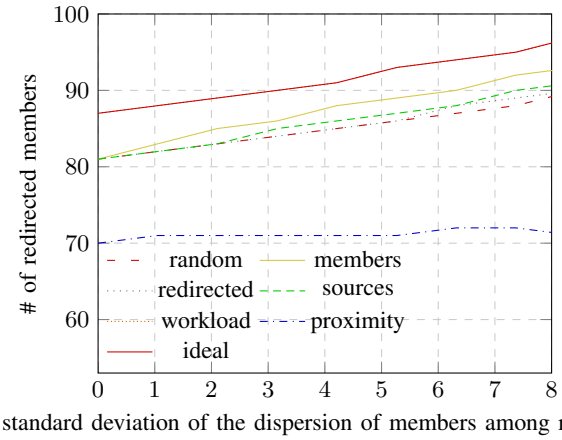


(a) Overlay with 10 reflectors and 100 members

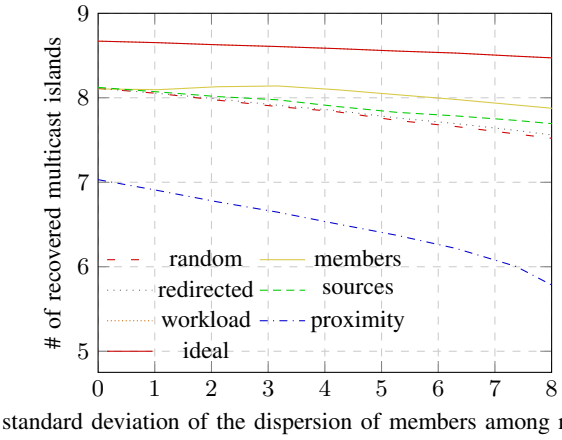


(b) Overlay with 10 reflectors and 100 members

Figure 4: Maximum number of redirected members according to network conditions when varying (a) the average bandwidth in the reflectors assuming a standard deviation of 4 Mbps and (b) the bandwidth standard deviation assuming an average bandwidth of 15 Mbps



(a) Redirected members



(b) Recovered multicast islands

Figure 5: Maximum number of redirected members (a) and multicast islands (b) when varying the initial dispersion of overlay members among multicast islands

rest of heuristics considering the roles of members or their distribution is better.

Figure 5b shows that the dispersion of members steadily decreases the number of recovered multicast islands. If a reflector is serving a number of members higher than the average, it is more likely that this reflector is serving several sources concurrently, reducing its capacity to cope with redirected members. For this reason, the sources heuristic performs better in these tests than the redirected heuristic. However, the sources heuristic may penalize a reflector when there are few reflectors and the reflector is the only one not supporting sources, as it would receive all the redirected members. Thus, the members heuristic performs better than the sources heuristic. It redirects members to the reflector supporting the least number of members, so the likelihood of supporting sources for this reflector is also low.

According to the results shown in the previous figures, the workload heuristic is the best choice in most of the tests. The proximity heuristic only performs well on overlays with few members per multicast island. Regarding heuristics based on the distribution of members among the overlay or their roles, all of them have quite similar performance. In contrast, while the members heuristic performs poorly with a massive number of multicast islands, its performance is noticeably more efficient than the rest in unbalanced overlays.

6 Conclusions

Several *on event* heuristics to perform self-stabilization in overlays have been presented. The performance of the heuristics has been compared using a full-mesh overlay tailored to support real-time interactive activities deployed on multiple multicast islands and controlled by a central entity.

The proposed heuristics are used to stabilize the overlay by reorganizing members when failures are detected. This reorganization implies building new data links among members of the overlay. The performance of each heuristic is measured using the network resources consumed by each reflector of the overlay after the self-stabilization process. Results are especially relevant when designing and implementing self-stabilization strategies in overlays

for real-time interactive activities among several groups of participants, such as in corporate e-training environments.

The workload heuristic achieves performance quite close to the ideal. The benefit of using this heuristic is higher as the number of members increases. In fact, the workload heuristic is a good approximation to the ideal, but simpler to implement due to the lack of dynamic bandwidth calculations. In contrast, the performance of the proximity heuristic is poor in most of the cases, and its use should be restricted to scenarios with few members per multicast island.

The rest of heuristics, based on the distribution of members among the overlay or their roles, achieve similar performance. However, the heuristic based on the number of members per multicast island is a good choice only when there is an initial imbalance of overlay members among multicast islands. In other scenarios, its performance decreases since it does not consider circumstances compromising the overlay such as the number of unicast connections and data sources.

Future work will be focused on measuring the quality of experience perceived by users during the self-stabilization process.

References

Alima, L., Haridi, S., Ghodsi, A., El-Ansary, S. and Brand, P. (2004) 'Self-properties in distributed k-ary structured overlay networks', *1st International Workshop on Self-* Properties in Complex Information Systems (Self-Star 2004)*, Bertinoro, Forlì-Cesena, Italy, pp. –4.

Baresi, L., Guinea, S. and Saeedi, P. (2013) 'Self-managing overlays for infrastructure-less networks', *IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2013)*, Washington, DC, USA, pp. 81–90. DOI: 10.1109/SASO.2013.25

Berns, A. and Ghosh, S. (2009) 'Dissecting self-* properties', *3rd IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2009*), Los Alamitos, CA, USA, vol. 0, pp. 10–19. DOI: 10.1109/SASO.2009.25

Di, P., Wählisch, M. and Wittenburg, G. (2010) 'Modeling the network layer and routing protocols', *Modelling and Tools for Network Simulation*, ed. Wehrle, K., Günes, M. and Gross, J., pp. 359–384, Springer, Berlin. DOI: 10.1007/978-3-642-12331-3_16

Dolev, S. (2000) *Self-stabilization*, MIT Press, Cambridge, MA, USA.

Draves, R., Padhye, J. and Zill, B. (2004) 'Comparison of routing metrics for static multi-hop wireless networks', *ACM 10th International Conferenfece on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM 2004)*, New York, NY, USA, pp. 133–144. DOI: 10.1145/1015467.1015483

El-Sayed, A., Roca, V. and Mathy, L. (2003) 'A survey of proposals for an alternative group communication service', *IEEE Network*, vol. 17, no. 1, pp. 46–51. DOI: 10.1109/MNET.2003.1174177

Ganek, A. G. and Corbi, T. A. (2003) 'The dawning of the autonomic computing era', *IBM Systems Journal*, vol. 42, no. 1, pp. 5–18. DOI: 10.1147/sj.421.0005

Gau, V., Wu, P. J., Wang, Y. H. and Hwang, J. N. (2009) 'Peer-to-Peer Streaming Systems', in Li, Q. and Shih, T. K. (Eds.), *Ubiquitous Multimedia Computing*, CRC Press, Florida, FL, USA, pp. 3–38.

Granda, J. C.,Nuño, P., Suárez, F. J. and Pérez, M. A. (2013) 'E-pSyLon: A synchronous e-learning platform for staff training in large corporations', *Multimedia Tools and Applications*, vol. 66, no. 3, pp. 431–463. DOI: 10.1007/s11042-012-1061-9

Granda, J. C., Nuño, P., García, D. F, and Suárez, F. J. (2015) 'Autonomic platform for synchronous e-training in dispersed organizations', *Journal of Network and Systems Management*, vol. 23, no. 1, pp. 183–209. DOI: 10.1007/s10922-013-9290-4

Haydarlou, A., Overeinder, B., Oey, M. and Brazier, F. (2006) 'Multi-level model-based self-diagnosis of distributed object-oriented systems', *Autonomic and Trusted Computing*, ed. Yang, L. T., Jin, H., Ma, J. and Ungerer, T., vol. 4158, pp. 67–77, Springer, Berlin. DOI: 10.1007/11839569_7

Hosseini, M., Ahmed, D.T., Shirmohammadi, S. and Georganas, N. D. (2007) 'Survey of application-layer multicast protocols', *IEEE Communications Surveys & Tutorials*, vol. 9, no. 3, pp. 58–74. DOI: 10.1109/COMST.2007.4317616

Kephart, J. O. and Chess, D. M. (2003) 'The vision of autonomic computing', *Computer*, vol. 36, pp. 41–50. DOI: 10.1109/MC.2003.1160055

Murase, T., Shimonishi, H. and Murata, M. (2006) 'Overlay network technologies for QoS control', *IEICE Transactions on Communications*, vol. 89-B, no. 9, pp. 2280–2291. DOI: 10.1093/ietcom/e89-b.9.2280

Nuño, P., Granda, J. C., Suárez, F. J. and García, D. F. (2013) 'Self-* in multimedia communication overlays', *Computer Communications*, vol. 36, no. 7, pp. 817–833. DOI: 10.1016/j.comcom.2012.12.009

Pianese, F., Perino, D., Keller, J. and Biersack, E. (2007) 'PULSE: An adaptive, incentive-based, unstructured P2P live streaming system', *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1645–1660. DOI: 10.1109/TMM.2007.907466

Saia, J. and Trehan, A. (2008) 'Picking up the Pieces: Self-healing in reconfigurable networks', *IEEE 9th International Symposium on Parallel and Distributed Processing (IPDPS 2008)*, Miami, FL, USA, pp. 1–12. DOI: 10.1109/IPDPS.2008.4536326

Sterritt, R. (2005) 'Autonomic computing', *Innovations in Systems and Software Engineering*, vol. 1, no. 1, pp. 79–88. DOI: 10.1007/s11334-005-0001-5

Takahashi, K., Mori, T., Hirota, V., Tode, H. and Murakami, K. (2013) 'A resilient forest-based application level multicast for real-time streaming', *IEICE Transactions on Communications*, vol. 96-B, no. 7, pp. 1874–1885. DOI: 10.1587/transcom.E96.B.1874

Tan, S., Waters, G. and Crawford, J. (2006) 'A performance comparison of self-organising application layer multicast overlay construction techniques', *Computer Communications*, vol. 29, no. 12, pp. 2322–2347. DOI: 10.1016/j.comcom.2006.02.020

Zhang, X., Liu, J., Li, B. and Yum, T. S. P. (2005) 'CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming', *IEEE 24th International Conference on Computer Communications (INFOCOM'05)*, Miami, FL, USA, vol. 3, pp. 2102–2111. DOI: 10.1109/INFCOM.2005.1498486