Situated vs. Global Aggregation Schemes for Autonomous Management Systems

Rafik Makhloufi, Guillaume Doyen, Grégory Bonnet, Dominique Gaïti

Situated vs. Global Aggregation Schemes for Autonomous Management Systems

Rafik Makhloufi*, Guillaume Doyen*, Grégory Bonnet† and Dominique Gaïti*
*ICD/ERA, UMR 6279. Université de Technologie de Troyes
†GREYC/MAD UMR 6072. Université de Caen Basse-Normandie

***Abstract*—In the context of autonomous network management, the Autonomic Managers (AMs) need to collect management information from other elements in order to infer an overall state of the network considered by the decision making process. Two concurrent strategies are commonly used to achieve this operation. On one hand, approaches based on a situated view only gather information in a bounded neighborhood, thus providing a high reactivity to AMs for control operations. On the other hand, approaches based on a global view provide a good accuracy at the cost of a larger convergence time. Being able to choose the best approach in a given context is crucial to ensure the efficiency of an autonomous management system. Thus, in this paper, we perform an exhaustive performance analysis of these approaches by considering typical schemes of both of them, namely a one-hop and two-hops situated view against gossip- and tree-based global aggregation schemes. Metrics we consider are the convergence time, communication and computation cost, scalability and the accuracy of estimated aggregates. Given them, we show under which conditions an approach outperforms the others.**

***Index Terms*—Autonomous Networking, Decentralized Aggregation, Situated View, Management Information.**

I. Introduction

The Autonomic Managers (AMs) of an autonomous management system need to collect management information from the network elements in order to infer an overall state of the network for the decision making process. Thus, the performance of the management system is directly depending on the quality of collected information that must meet some constraints such as accuracy, consistency and availability. This information is collected through aggregation schemes according to a situated view (SV) where each node has the knowledge of a subset of the network nodes or according to a global view (GV) where global aggregates are computed on each node to infer the overall state within the network.

Previous studies show that each aggregation scheme is efficient in a given context. For example, gossip schemes are less sensitive to faults and dynamics than tree ones, but they need more communication, computation and time to converge. Thus far, the existing evaluations on the aggregation schemes only propose to compare tree-based and gossip-based schemes. They do not include the situated schemes in their comparisons. So, we do not know how this technique behaves in comparison to the global schemes. Thereby, there is a need to study the performance of these aggregation categories in order to learn exactly when using each of them.

In this paper, we propose a comparative study of the performance of situated and global aggregation schemes. For this we implement three typical aggregation schemes, one from the situated view with two global ones, a gossip and a tree aggregation schemes. Then, we compare them according to standard evaluation criteria that are convergence time, computation and communication costs, scalability and accuracy.

This paper is organized as follows. We first present the related work on the evaluation of aggregation schemes in Section II. We give an overview of the existing global and situated aggregation schemes and we describe the aggregation schemes that we have implemented from each category in Section III. Subsequently, we present our evaluation of the developed schemes in Section IV. Finally, we conclude and we present our perspectives in Section V.

II. Related Work

Because of the emergence of several decentralized aggregation protocols, many studies have been performed in order to compare their performance.

Bawa *et al.* [1] propose a set of aggregation schemes for estimating basic aggregates on a P2P network. They compare one gossip-based scheme Propagate2All to two tree-based schemes: SingleTree and MultipleTree. This study shows that the tree outperforms the gossip in terms of time, communication and computation costs, but the latter is more accurate under churn. The authors compare these global schemes, but do not discuss the situated view in their comparison.

Wuhib *et al.* [2] present G-GAP (Gossip-based Generic Aggregation Protocol), a gossip protocol for continuous monitoring of aggregates, where the tradeoff between the estimation accuracy and the overhead can be controlled. G-GAP is an extension of the push-synopses scheme of [8]. The authors compare G-GAP to GAP (Generic Aggregation Protocol), a tree-based aggregation protocol that we describe in the next section. Contrary to the first presented study, this evaluation shows that GAP outperforms the gossip protocol for comparative overhead, both in terms of accuracy and robustness.

Birman [3] discusses the strengths and limitations of gossip schemes. On one hand, the author presents their advantages: simplicity, bounded load on nodes, topology independence and robustness to transient network disruptions. On the other hand, according to him, the small bounded message sizes and the relatively slow periodic exchanges limit the information carrying

capacity of gossip. Furthermore, gossip scales well in some dimensions but not for all. Gossip is also a community process where all the nodes are dependent upon the correct behavior of all other nodes. Therefore, a malicious or malfunctioning node can delay or even defeat the aggregation. This paper does not provide quantitative comparison results, but only a qualitative analysis of the gossip's limitations and strengths.

In our previous work [4], we presented an overview of a set of decentralized aggregation schemes and provided a multi-criteria classification of them. The provided theoretical comparison results of this study were collected from the literature. These results are limited, since the original experiments are performed under different test conditions.

To summarize, according to these studies, gossip schemes ensure fault-tolerance. However, the large number of exchanged messages causes more communication and computation overhead than tree-based schemes. Thereby, tree-based schemes execute themselves in a better convergence time and a lower communication and computation cost due to their optimization of the number of exchanged messages on the tree. However, their hierarchical structure with a unique path between each node and the root lets hierarchical schemes be more sensitive to faults than the gossip ones.

Globally, these evaluation studies show that each of the aggregation categories is better than the other one in a given context. All these studies consider only global aggregation schemes. To the best of our knowledge there is no work in the literature that compares situated approaches with global ones. Thus, we do not know the performance of the situated schemes in comparison to the global ones. So, it is necessary to clearly identify when we need to use each of these aggregation categories for collecting aggregates.

III. Compared aggregation schemes

We develop and we implement three typical and representative aggregation schemes inspired from existing ones. In this section, we give a brief overview of them and we present the algorithms we implemented in our study.

A. Global view

According to the global view, aggregates can be collected over a tree or through gossip.

1) Tree: This involves the use of a hierarchical structure for collecting aggregated management information. The computation of aggregates is done hierarchically in a bottom-up fashion. The aggregation algorithm converges when the computed global aggregate is available on the root of the tree [1].

In this category, we implemented a push tree-based aggregation scheme that is a combination of GAP [5] and the deployment topology proposed in [6]. This scheme consists in a structured P2P overlay where all nodes communicate their local aggregates through a DHT to a single root node. The latter computes an overall aggregate and uses a publish-subscribe mechanism to spread it on all nodes that subscribed to the diffusion group concerning the monitored variable, as shown in this algorithm. A node does not know in advance the tree structure and its children, but it discovers it when messages are exchanged.

Algorithm 1 Push tree scheme executed by a node i

(a) Active thread	(b) Passive thread
1: $p \leftarrow$GetParent()	1: **loop**
2: send $(i,(X_i,1))$ to p	2: msg$\leftarrow$receive $(j,(X_j,w_j))$
3: msg$\leftarrow$receive(j,X_j)	3: $state_i \leftarrow$update(msg)
4: $state_i \leftarrow$update(msg)	4: $p \leftarrow$GetParent()
	5: send $(i,(X_i,w_i))$ to p
	6: **if** i is root **then**
	7: wait until receive all aggregates
	8: diffuse $(i, \frac{X_i}{w_i})$
	9: **end if**
	10: **end loop**

As illustrated in Algorithm 1, each node executes two different threads: an active and a passive one. The active thread (Algorithm 1.a), executed once on a node i, initiates the information exchange. The passive thread (Algorithm 1.b) waits for messages (msg) sent by an initiator to process them. Initially, each node i uses the GetParent() method to select its parent (line a.1) and sends it a couple $(i,(X_{raw_i},1))$ including its $NodeId$, its raw value and its weight (line a.2). A node i that receives a message from a child j (line b.2), updates its local state over the update(msg) method (line b.3) where it calculates a new partial aggregate through those of all its children. It then forwards the new aggregate in a pair $(i,(X_i,w_i))$ to its parent (lines b.4 and b.5). If node i is the root (line b.6) then it waits until it receives all its children's aggregates (line b.7), and it diffuses the global aggregate X_i over a publish-subscribe system on all the subscribed nodes (line b.8). Thus, each node that receives X_i (line a.3) updates its partial aggregate with the global one (line a.4).

2) Gossip: Unlike tree-based techniques, where nodes are organized into a tree, gossip-based schemes do not require a particular structure to perform aggregation. At each round of the aggregation process, a node contacts one or more of its neighbors usually chosen randomly and exchanges information with them [3], [7], [8]. Initially in the network, each node has only its own raw management information. The aggregation algorithm converges when the computed global aggregate is available across all the network nodes.

The aggregation scheme developed here and presented in Algorithm 2 is based on the push-pull gossiping scheme [9] with symmetric information exchanges where both nodes send and receive their estimates.

As illustrated in Algorithm 2, node i calls the GetNeighbors(1) method to select uniformly at random one node j from the list of its direct neighbors $\mathbb{D}_i$ which is obtained over the entire set of networks nodes (line a.2). Then, i sends to j a message (i,X_i) containing its local aggregate and waits for a response with the remote node j (line a.3). When it receives a couple (j,X_j) from j (line a.4), it updates its local state through the update(msg) method that computes a new partial aggregate according to the selected aggregate function

Algorithm 2 Push-pull gossip scheme executed by node i

(a) Active thread

1: **loop**
2: $j \leftarrow$GetNeighbors(1)
3: send (i, X_i) to j
4: msg$\leftarrow$receive(j, X_j)
5: $state_i \leftarrow$update(msg)
6: wait(round duration)
7: **end loop**

(b) Passive thread

1: **loop**
2: msg$\leftarrow$receive(j, X_j)
3: send (i, X_i) to j
4: $state_i \leftarrow$update(msg)
5: **end loop**

(line a.5). The node i repeats the same process at each round (line a.6). When the passive thread (Algorithm 2.b) of node i receives an exchange request message (line b.2), it replies with its local aggregate (line b.3) and then it updates its local state through the update(msg) method (line b.4).

B. Situated view

In a concurrent way, alternative decentralized management approaches like [10], [11] propose to limit the view of each node to some nodes by using a situated view. Thus, the knowledge of a node is limited to its direct neighbors or a part of the network nodes [12]. The size of this view is defined by a number of nodes or a number of hops.

We have implemented a typical situated scheme inspired from HyParView (Hyper Partial View) [10], where each node maintains a partial view of a part of network nodes, bounded by a maximum number of hops. A node i can then obtain an aggregate of its view by collecting management information from its h-hops neighbors.

Algorithm 3 Pull situated view scheme on node i

(a) Active thread

1: $\mathbb{D}_i \leftarrow$GetNeighbors(all)
2: send (i, h) to $\mathbb{D}_i$
3: msg$\leftarrow$receive(j, X_{raw_j})
4: $state_i \leftarrow$update(msg)

(b) Passive thread

1: **loop**
2: msg$\leftarrow$receive(j, h)
3: send (i, X_{raw_i}) to j
4: $h \leftarrow h - 1$
5: **if** $h > 0$ **then**
6: $\mathbb{D}_i \leftarrow$GetNeighbors(all)
7: send(j, h) to $\mathbb{D}_i$
8: **end if**
9: **end loop**

As shown in Algorithm 3, the requesting node i gets the list $\mathbb{D}_i$ of all its direct neighbors through the GetNeighbors(all) method (line a.1) and sends them a query message (i, h) (line a.2). Each node i that receives an aggregation request message (line b.2) verifies if it does not previously answer to the same request coming from j in the same aggregation cycle. If so, i answers by sending its ID and its local raw value (i, X_{raw_i}) directly to the requesting node (line b.3). Then, the node i decrements the number of hops contained in the received message (line b.4). If the maximum number of hops is not reached (line b.5), then the node i forwards the received request (j, h) to all its direct neighbors (lines b.6 and b.7). When a requesting node i receives an answer (j, X_{raw_j}) from a neighbor (line a.3), it adds this pair of values to its maintained neighbors set $\mathbb{L}_i$ and updates its own state by computing a new partial aggregate (line a.4).

IV. Evaluation study

We present in this section our evaluation study of the performance of these situated and global aggregation schemes.

A. Experimental framework

We performed simulations in the context of the monitoring service under the testbed and the scenarios described below.

1) Testbed and Simulation scenarios: We conducted our evaluation in the FreePastry simulator, an open-source Java implementation of the Pastry DHT [13]. In order to provide realistic results, we carry out all our experiments with the Euclidean network topology model. We also rely for the tree-based scheme on Scribe [14] to spread the root's aggregates on all the subscribed nodes.

Based one the realistic parameters summarized in Table I, we run within a static network each of the developed aggregation schemes to compute an average of randomly generated values ranging between 0 and 100. The situated scheme is executed with a view limited to the direct neighbors (SV1) and also with two-hops (SV2). The gossip process is executed with rounds of duration 600ms that corresponds to the maximum time for an information exchange. We consider a network size varying from 2 to 1000 nodes. To give a sufficient statistical significance to the results, each value presented here is an average of the values obtained on 100 executions of the aggregation algorithms.

TABLE I
Simulation parameters

Parameter	Value
Aggregate function	Average
Network topology model	Euclidean
Topology maintaining frequency	200ms
Values changing frequency	20sec
Tolerated error (ϵ)	0
Neighborhood degree	8
Gossip round duration	600ms
Number of nodes (N)	[2;1000]
Number of hops in SV (h)	[1;2]

B. Evaluation results

Since in the literature, the acknowledged studied evaluation criteria for aggregation schemes are convergence time, communication and computation cost, scalability and accuracy [1], [2], [3], [4], we propose here to carry out a comparison of the developed schemes according to these criteria.

1) Convergence time: It is the elapsed time for both the communication and the computation of a global aggregate [1]. Therefore, it is the necessary time between the initialization of the aggregation process and the time t when all nodes hold the aggregation results. Thereby, T_{conv}=T_{agg}-T_{init} (equations 1, 2 and 3). Thus, in the case of global view, T_{conv} corresponds to the time when all nodes hold the same global aggregate. In the situated scheme, this condition cannot be reached because each node retrieves only the values of its h-hops neighbors. So, we

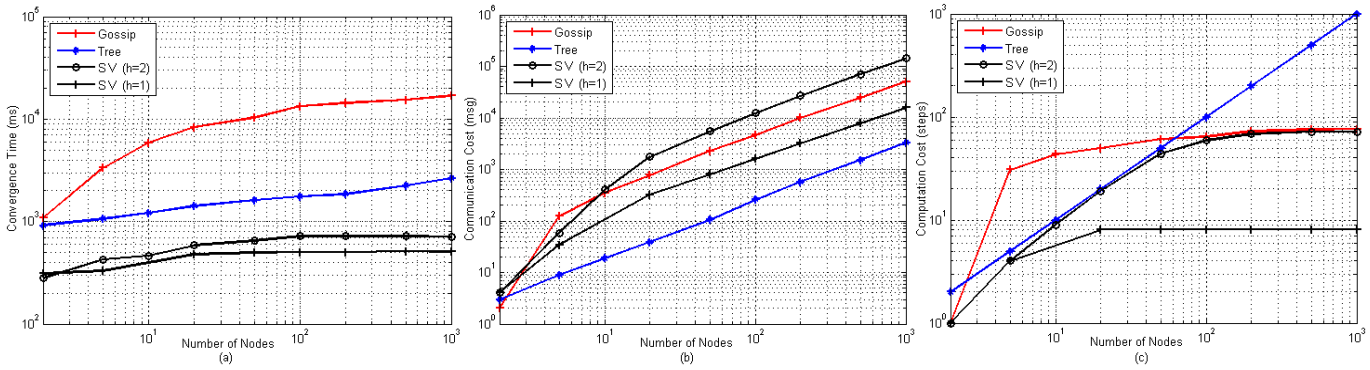

Fig. 1. Simulation results on: (a) Convergence time; (b) Communication cost; (c) Computation cost [log-log scale]

measure the required time for each node to collect information from its neighbors and to calculate a partial aggregate.

$$T_{init} = t \quad , \quad \forall i \in \{1, ..., N\},\ X_i^t = X_{raw_i} \qquad (1)$$

$$T_{agg}(GV) = t \quad , \quad \forall i, j \in \{1, ..., N\},\ |X_i^t - X_j^t| < \epsilon \qquad (2)$$

$$T_{agg}(SV) = t \quad , \quad \forall i, \forall j \in \{1, ..., K\},\ X_{raw_j} \in \mathbb{L}_i \qquad (3)$$

We observe in Figure 1.a a large convergence time for the gossip scheme followed by the tree, then a relatively low time for the situated view. Under a network of 1000 nodes, the gossip's convergence time is about 6 times higher than the one of the tree and about 23 times the one of SV2. This high delay is explained by the blind communication used to exchange messages at each round of the gossip. For the tree, the convergence time is the delay required to send all the values to the root node and to spread the computed aggregate on all the subscribed nodes. The situated scheme requires a low time to converge because at one time each node sends simultaneously one request message to all its direct neighbors and then computes a partial aggregate of the received values to two-hops nodes. This time is lower in SV1 when only direct neighbors are contacted. Thus, in terms of convergence time, the situated scheme scales better and converges more quickly than the global ones.

2) Communication cost: The communication cost [1] is the sum of sizes of messages sent between any node pairs (i, j) during the aggregation process. The communication cost for the developed algorithms is considered as the number of messages sent by all nodes because all messages have approximatively the same size, in the sense that each message contains only two or three numerical values. Thus, $C_{comm} = \sum_{i=1}^{N} C_{comm_i}$, where C_{comm_i} is the the number of messages sent by node i and N is the number of network nodes.

In Figure 1.b, we see that the communication cost is proportional to the number of nodes. Under a network of 1000 nodes, the communication cost of SV2 is almost 3 times higher than the one obtained in the case of gossip and about 42 times the one of the tree. This high communication cost is explained by the fact that the latter is based on a broadcast algorithm where each node floods its request message on all its h-hops neighbors. SV1 causes more communication overhead than the tree and less than the gossip. In the latter, a node exchanges its value with only one other node, so it causes about 9 times less overhead than SV2. The tree scheme causes the lowest communication overhead that corresponds to the messages sent in a bottom-up fashion to the root and those used by Scribe to spread the global aggregate. For the gossip scheme, it involves more messages to converge than the tree because it uses a blind communication over multiple rounds. Thus, SV2 involves more messages exchanges than the aggregation through the global schemes. This overhead can be reduced by limiting the view size of nodes to the direct neighbors.

3) Computation cost: The computation cost [1] is the maximum computation cost among all the nodes in the network. For a single node, the computation cost is the number of steps taken by the aggregation process that is executed on the node. Thus, $C_{comp} = \max(C_{comp_i}), i \in \{1, ..., N\}$.

We notice in Figure 1.c that when we have a large number of nodes, the computation cost of the situated view is less important than the tree one and less than the gossip one. In the latter, exactly one update operation is executed on a node in each round. For the situated scheme, the computation cost depends on the view size of each node, since in one round an update operation is executed at each reception of a neighbor's value. We observe a higher computation cost for the tree-based scheme because according to the developed algorithm, the worst case is registered at the root node where the computation cost is equal to the number of nodes. A lower computation cost was expected which is not the case. It is due to the reactive mechanism in which each node of the tree directly sends the received messages to its parent. Thus, a node does not wait to receive messages from all its children before sending the computed partial aggregate.

The situated view causes less computation overhead than the the global schemes. It is also more scalable than them, according to this criterion.

4) Accuracy of estimated aggregates: When messages are exchanged between nodes, the initial average is redistributed among them. Thus, the aggregation process does not change the global average but it decreases the variance over the set

of all estimates in the system. Thus, in order to show the distribution of estimates over all the network nodes and to show how far these values lie from the average value, we compute the variance over all the estimates (equation 4).

$$V(X) = \frac{1}{N}\sum_{i=1}^{N}(X_i - \overline{X})^2. \quad (4)$$

In order to evaluate the accuracy of estimated aggregates for each aggregation scheme, we fix the size of the network to 1000 nodes and we measure the variance over the partial nodes's aggregates after each cycle of duration 200ms.

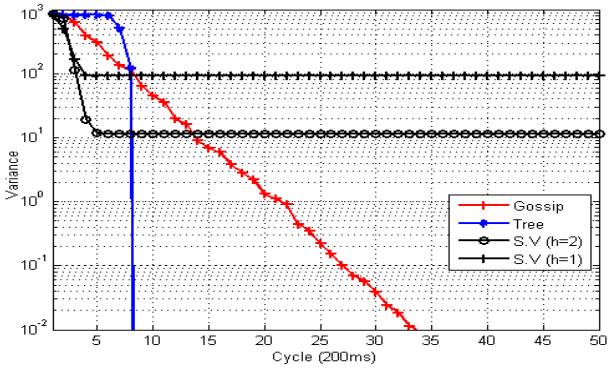

Fig. 2. Accuracy of estimated aggregates [semi-log scale];

We see in Figure 2.a that the variance between the distributed aggregates decreases with an increase in the number of cycles. For the situated scheme, the minimal variance is always greater than 0. Thus, it is less accurate than the global schemes. This is caused by the computation of partial aggregates on each node, contrary to gossip or tree where a global aggregate is computed. When we restrict the view size in the situated view to the direct neighbors of nodes, this minimizes the overhead and the convergence time. Although, we get less accuracy on the estimation of global aggregates. Thus, the global schemes ensure more accuracy than the situated one in the aggregation.

Globally, the two-hops situated scheme involves less convergence time and less computation cost than the global ones. It is also more scalable than them regarding these criteria, with a comparable communication cost. But, it provides less accuracy in the estimation of aggregates than the global schemes. Thus, it is more preferable to use the global schemes when we need more accuracy and exactitude and to use the situated one when we need to reduce the time, communication and computation costs. When we limit the view size of nodes to the direct neighbors, we reduce the different costs of the situated scheme but we lose in termes of the accuracy of the estimated aggregates. Moreover, concerning the global schemes, one can note that we obtained consistent and comparable evaluation results with those presented in the related work for the common criteria.

V. Conclusion and Future Work

This paper compares situated aggregation schemes to the global ones. It provides quantitative results on the use of these schemes for collecting aggregates. We evaluate the performance of each scheme according to convergence time, communication and computation costs, scalability and the accuracy of estimated aggregates. Through the obtained simulation results, we confirm that none of the protocols is better than another. Their performance depends on the context in which they are deployed. The situated scheme outperforms both gossip and tree in terms of convergence time, computation cost and scalability. However, for the accuracy of estimated aggregates, the global schemes outperform the situated one. Finally, the communication cost of the situated view depends on the view size of nodes.

In an effort to enhance this study, we are currently working on the establishment of realistic models to represent the level of information dynamics and network dynamics. This will allow us to evaluate the fault-tolerance of each scheme by measuring the impact of both the network and information dynamics on the decision making process quality. We also plan to pursue this work by designing an adaptive management system able to combine the use of global and situated schemes by selecting the suitable scheme to use according to the current context of the management information and its environment.

References

[1] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani, "Estimating aggregates on a Peer-to-Peer network," Tech. Rep., 2003.

[2] F. Wuhib, M. Dam, R. Stadler, and A. Clem, "Robust monitoring of network-wide aggregates through gossiping," *TNSM*, vol. 6, no. 2, pp. 95–109, 2009.

[3] K. Birman, "The promise, and limitations, of gossip protocols," *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 5, pp. 8–13, 2007.

[4] R. Makhloufi, G. Bonnet, G. Doyen, and D. Gaïti, "Decentralized aggregation protocols in peer-to-peer networks: a survey," in *MACE*, 2009, pp. 111–116.

[5] A. G. Prieto and R. Stadler, "A-GAP: an adaptive protocol for continuous network monitoring with accuracy objectives," *TNSM*, vol. 4, no. 1, pp. 2–12, 2007.

[6] R. Makhloufi, G. Bonnet, G. Doyen, and D. Gaïti, "Towards a P2P-based Deployment of Network Management Information," in *AIMS*, 2010, pp. 26–37.

[7] M. Dietzfelbinger, "Gossiping and broadcasting versus computing functions in networks," *Discrete Applied Mathematics*, vol. 137, no. 2, pp. 127–153, 2004.

[8] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *FOCS*, 2003.

[9] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *TOCS*, vol. 23, no. 3, pp. 219–252, 2005.

[10] J. Leitao, J. Pereira, and L. Rodrigues, "Large-Scale Peer-to-Peer Autonomic Monitoring," in *DANMS*, 2008, pp. 1–5.

[11] D. F. Macedo, A. L. dos Santos, G. Pujolle, and J. M. S. Nogueira, "MANKOP: A Knowledge Plane for wireless ad hoc networks," in *NOMS*, 2008, pp. 706–709.

[12] T. Bullot, R. Khatoun, L. Hugues, D. Gaïti, and L. Merghem-Boulahia, "A situatedness-based knowledge plane for autonomic networking," *Int. J. Netw. Manag.*, vol. 18, no. 2, pp. 171–193, 2008.

[13] A. I. T. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," in *Middleware*, 2001, pp. 329–350.

[14] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "Scribe: a large-scale and decentralized application-level multicast infrastructure," *JSAC*, vol. 20, no. 8, pp. 1489–1499, 2002.