INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT
Int. J. Network Mgmt (2014)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nem.1875

A survey and performance evaluation of decentralized aggregation schemes for autonomic management

Rafik Makhloufi,[1,*†] Guillaume Doyen,[2] Grégory Bonnet[3,4,5] and Dominique Gaïti[2]

[1]*OPTIWAYS, 21 route de la Croix, 78110 Le Vésinet, France*
[2]*ICD/ERA, UMR 6279, Université de Technologie de Troyes, Troyes Cedex, France*
[3]*Normandie Université, Caen, France*
[4]*UNICAEN, GREYC, Caen, France*
[5]*CNRS, UMR 6072, Caen, France*

SUMMARY

Over the last few years, autonomic network and service management has emerged as a serious alternative to traditional management approaches. In these systems, distributed entities, called autonomic managers, perform monitoring and control operations in an autonomous and decentralized way. The monitoring consists of providing indicators on the state of the system. Several monitoring solutions have been proposed to enable autonomic managers to obtain a partial or complete knowledge of an indicator through aggregation processes. Such a profusion of solutions raises important questions regarding the choice of an aggregation scheme in a particular operational context and for a particular management information because each class of solution presents different benefits and weaknesses. That is why, in this paper, we present the result of our study of decentralized aggregation schemes for autonomic network and service monitoring. The contribution is twofold: (i) a survey of decentralized aggregation schemes based on a refined taxonomy; and (ii) the results of an evaluation campaign we performed to compare typical aggregation schemes. These results highlight the context, in terms of the managed network behaviour as well as information dynamics, in which each aggregation scheme outperforms the others, thus helping autonomic management system designers in choosing the best scheme for their management purpose. Copyright © 2014 John Wiley & Sons, Ltd

Received 8 January 2013; Revised 2 June 2014; Accepted 19 July 2014

1. INTRODUCTION

Over the last few years, numerous approaches have emerged in order to manage networks and software services in a decentralized and autonomous way. Initiatives such as autonomic computing [1,2] or organic computing [3–5] proposed, even through by different means, to build autonomic systems that are self-configuring, self-healing, self-protecting and self-optimizing. In these autonomic systems, autonomic managers (AMs) are distributed management entities that cooperate to perform monitoring and control operations. If traditional management frameworks rely on a centralized architecture in which information is gathered in a single point and decisions are taken by a central authority, autonomic management frameworks distribute the management operations to AMs that are potentially embedded in all managed elements, leading to fully decentralized architectures [6,7]. Although different levels of distribution are possible, fully decentralized approaches are the most promising ones regarding their scalability and resilience while being the most challenging in terms of performance and complexity. Indeed, in that context, all management operations that can be divided into monitoring and control have to be reconsidered.

*Correspondence to: Rafik Makhloufi, CERMICS/SOWG, Ecole Nationale des Ponts et Chaussées, Cité Descartes 77455, Marne-la-Vallée Cedex 2, France.
†E-mail: rafik.makhloufi@univ-reims.fr

Monitoring is in charge of providing information about the managed system (network or software service) to control processes embedded in AMs to form local control loops. This information can be raw data extracted from the local managed element through a local instrumentation (or from a remote one), or a global indicator resulting from the aggregation of data coming from different managed elements. As a consequence, if the monitoring operations performed by traditional centralized architectures only consist of the collection of information, monitoring in autonomic management frameworks means collecting information and also disseminating it to all interested AMs. Thus, it extends the functionalities of monitoring from only the collection to both the collection and dissemination. Considering both functionalities for aggregated information is even more challenging in terms of accuracy, consistency and availability of monitored information, as they induce distributed operations that potentially imply all AMs and that are critical because their quality of service impacts the overall performance of the management system.

To this end, numerous decentralized schemes have been proposed to build scalable, robust and accurate distributed monitoring solutions. For example, on the one hand, situated views (SVs) [8], in which each node has the knowledge of a subset of the network nodes, have been proposed to build highly reactive autonomic management systems. On the other hand, global views (GVs), where global aggregates are computed and disseminated on each managed element to infer the overall state of the network, have emerged as an extension of traditional centralized solutions, thus bringing more accuracy than SVs at the cost of larger convergence time.

As a result, management system designers aiming to deploy autonomic monitoring infrastructures are faced with numerous solutions, each presenting its own benefits and drawbacks. For example, gossip schemes are less sensitive to faults and dynamics than tree-based schemes and enable both the collection and dissemination of information, but they need more communication, computation and time to converge. Moreover, to the best of our knowledge, no study comparing each class of solution in the same generic context has been performed to date. Indeed, the performance evaluation of schemes has been performed by their authors as partial proof of their proposal validity. As a consequence, evaluation contexts (e.g. metrics, range of values, impacting factors and scenario) are different from one evaluation to another, making a straight comparison of proposals hard to establish.

In this paper, we propose a comparative study of the performance of situated and global schemes that can be considered for the monitoring operation of autonomic management frameworks. Within it, we especially focus on aggregation rather than dissemination because it is the most challenging owing to its composition of both distributed computing and communication aspects. One can note that some of the schemes we consider are not restricted to autonomic management and can also be used in standard monitoring frameworks by centralized control entities or by the human administrator for reporting purposes.

Our contribution is twofold. The first contribution is a survey of current decentralized aggregation schemes that can be considered as the monitoring operation of autonomic management systems. It proposes a refined taxonomy that classifies them according to the network structure they rely on, their propagation technique, the view they provide and their use of neighbourhood information in communication. The second contribution is a performance evaluation. To that aim, we implemented three representative aggregation schemes: the first one is situated, the second one is global and gossip based and the third one is global and tree based. We have compared them according to standard evaluation criteria that are convergence time, computation and communication costs, scalability and accuracy, and we show in which operational context and for which kind of information a scheme outperforms the others. Such an evaluation is intended to provide guidelines for autonomic management system designers towards the choice of an approach given the nature of the monitored information and the operational context it will be deployed on. Because such a study could consider numerous criteria leading to an explosion in the number of collected results, we have defined a perimeter that, while intended to cover a wide range of applicability, bounds and limits our results. These are: (i) the use of an autonomic management scenario built upon an overlay of uniformly distributed nodes; (ii) the use of a linear model as an abstraction of management information; and (iii) the sole consideration of quantitative metrics as evaluation parameters.

The paper gathers in a single place the work already presented by Makhloufi et al. [9–11]. Their first paper [9] surveys the current approaches for decentralized aggregation in autonomic monitoring;

DOI: 10.1002/nem

their second paper [10] presents a comparison of global versus situated schemes in a static context; and their third paper [11] evaluates the impact of dynamics on aforementioned aggregation schemes. Also, it extends this previous work by providing an extended state of the art of aggregation protocols and by enhancing the analysis of their performance up to the qualitative analysis of the context in which one protocol outperforms the others.

The paper is organized as follows. We first present the related work on the evaluation of decentralised aggregation schemes in Section 2. We give an overview of aggregation functions, and we propose a refined taxonomy for these schemes in Section 3. Then, we survey the existing global and situated proposals in Section 4. We describe the aggregation schemes that we have implemented from each category in Section 5. Subsequently, we present our evaluation results in Section 6. Finally, we conclude and present our perspectives in Section 7.

2. RELATED WORK

Because of the emergence of numerous decentralized aggregation protocols, many studies have been performed in order to compare their performance [12–16] in the context of network management. Other surveys have been performed in the context of wireless sensor networks [17–27].

Bawa *et al.* [12] proposed a set of decentralized aggregation schemes for estimating basic aggregates on a peer-to-peer (P2P) network. They compared one gossip-based aggregation scheme, Propagate2All, that uses a diameter for denoting the upper bound to which the network is known, with two tree-based aggregation schemes, which are as follows: (i) SingleTree, where a node broadcasts a query to build a spanning tree on the network; and (ii) MultipleTree, an enhancement of SingleTree that creates several independent spanning trees rooted at the querying node. 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. If this work is a first step towards the establishment of criteria that lead to the choice of a scheme for an aggregation purpose, their use of a static and dedicated topology (snapshot of the Gnutella network) restricts the applicability of the results to their own operational context. Also, the authors only compared global schemes and did not include the situated schemes in their comparison, while it is commonly used in autonomic network management.

Wuhib *et al.* [13] presented gossip-based Generic Aggregation Protocol (G-GAP), a gossip protocol for the continuous monitoring of aggregates where the trade-off between the estimation accuracy and the overhead can be controlled. G-GAP is an extension of the push-synopses scheme discussed in the work of Kempe *et al*. [28]. It tends to overcome the mass loss problem and renders the protocol robust against crash failures. The authors compared G-GAP with Generic Aggregation Protocol (GAP), a tree-based aggregation protocol that we describe in the next section. Contrary to the study of Bawa *et al*. [12], this evaluation shows that GAP outperforms the gossip protocol for comparative overhead, in terms of both accuracy and robustness, leading to the conclusion that, as compared with [12], evaluations bring different results according to the operational context they consider together with the tuning of schemes.

Birman [14] discussed the strengths and limitations of gossip schemes. On the one hand, the author presented their advantages: simplicity, rapid convergence, bounded load on nodes, topology independence, easy local information discovery and finally 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 a gossip algorithm. Furthermore, gossip scales well in some dimensions, but not for all. For example, a steadily increasing rate of events can exhaust the carrying capacity of the gossip information channel as the relatively slow rate of gossip can be an obstacle. Gossip is also a community process where all the nodes are dependent upon the correct behaviour of all other nodes. Therefore, a malicious or malfunctioning node can delay or even defeat the aggregation. The authors did not provide quantitative comparison results, but only a qualitative analysis of the gossip's limitations and strengths.

Jesus *et al.* [15] discussed a large set of aggregation protocols, considering tree-based and gossip-based protocols and some situated schemes with respect to their underlying aggregation function. They highlighted that tree-based protocols require a specific routing structure (e.g. spanning tree) to

DOI: 10.1002/nem

operate. Thus, those protocols are very cheap in terms of message exchanges but sensitive to churn and link failures. A solution consists in using other kinds of protocols that do not require such a specific network structure. SVs based on local neighbourhood are very fast, fault tolerant but unprecise. SVs based on random walk sampling are slower and less fault tolerant, but more precise for counting operations. Finally, gossip-based approaches are churn resilient, precise but very costly in terms of message exchanges. However, as in [14], this paper does not provide quantitative results, but only a qualitative analysis of the protocols' properties.

In the context of wireless sensor networks, some surveys [19,22,23,25,26] are out of the scope of this paper as they primarily focus on security and encryption schemes that can be embedded in aggregation protocols without discussing aggregation performances. The other surveys focus on aggregation protocol performance under resource consumption constraints [17,18,20,24,27]. The authors compared tree-based approaches, gossip-based approaches (called multi-path approaches) and cluster-based schemes. They also considered push-based schemes where the nodes are active participants and diffuse proactively their data to the sink and pull-based schemes where the nodes transmit their data when requested by the sink. Except for [17], those studies have compared the protocols on the basis of their original article without new experiments. These studies highlighted that the aggregation protocol performances are tightly coupled with the network infrastructure. For instance, [17] has shown that the latency caused by tree-based aggregation is proportional to the number of hops between the sink and the farthest source. All those studies agreed with the fact that tree-based and cluster-based protocols have proven to be effective in network topology management, energy minimization and data aggregation but are sensitive to churn and link failures. But even if these surveys highlighted the fact that cluster-based and tree-based approaches seem more interesting for wireless sensor networks because of their low energy cost, they did not provide quantitative results.

As a conclusion, current evaluation studies exhibit different performance results of aggregation schemes for autonomic monitoring. Such divergences are mainly due to the lack of a common evaluation framework that would enable a direct comparison of proposals. Moreover, the tuning of protocols to the context they are evaluated in also makes the identification of global properties and accurate deployment contexts important. Finally, while being commonly used for the monitoring in autonomic network management systems, the situated schemes have never been confronted to global schemes. This is why the research effort we present in this paper is: (i) a state-of-the-art aggregation scheme that can form the monitoring operation of autonomic management systems; and (ii) an evaluation of a generic version of standard schemes.

3. AGGREGATION SCHEMES

In this section, we briefly present and define, in a general way, the concepts related to the aggregation of data and propose a taxonomy for distributed aggregation schemes that can be used in a networking context (e.g. grids [29], P2P [30], mobile ad hoc networks or sensor networks [31]). In the next sections, we will focus our study on those that can be considered for autonomic management purposes.

3.1. Definition

Aggregation is a process that consists in combining several numerical values into a single representative value, and an aggregation function performs this operation [32]. The basic aggregate functions are counts, sums, averages, minima and maxima [33]. However, advanced aggregates can be computed from those simple aggregates such as approximated counts [34], histograms [35], parameter estimations [36,37], spectral analysis [38] or random linear projections [39].

As aggregation functions are mathematical functions, they are mainly centralized. However, in network management, network operators are interested in some statistical data allowing a decentralized monitoring of distributed resources such as free storage space, number of files shared or active nodes, loads on critical components and average lifetime of a node. Consequently, aggregation in network management is intended as a summarizing mechanism of the overall state within the network [40].

DOI: 10.1002/nem

It refers to a set of protocols that produces an indicator to evaluate a global property of a system. Hence, these functions allow a local and autonomous access to a global information in order to simplify the monitoring in distributed system. For example, an aggregate that reaches a specific value (i.e. threshold) may trigger the execution of some operations.

3.2. Classification of aggregate functions

As the amount of distributed data is becoming increasingly important owing to the large number of nodes in networks, there is a need to design an efficient and decentralized approach to manage these data [41]. Moreover, such approaches need to take into account the network domain specificities, such as redundant information or node dynamics. To this end, Madden *et al.* propose a classification of aggregate functions according to four properties [42] that we recall in a summarized way later in the text and that we illustrate in Table 1:

- *Duplicate sensitivity*: Duplicate-sensitive aggregates change when a duplicate reading is reported (e.g. when a node sends results to multiple parents, it can be counted multiple times).
- *Exemplary versus summary*: An exemplary aggregate returns one or more representative values from the set of all values, while a summary aggregate computes some properties over all values.
- *Monotony*: In monotonic aggregates, when two partial aggregates s_1 and s_2 are aggregated into s', we have $\forall s_1, s_2\, s' \geq \max(s_1, s_2)$ or $\forall s_1, s_2\, s' \leq \min(s_1, s_2)$.
- *Partial state*: It is the amount of information required for each partial aggregate record. For example, a partial *average* record consists of a pair of values, whereas a partial *count* record constitutes only a single value. There are five subcategories of aggregates in this dimension: distributive [43], algebraic [43], holistic [43], unique [42] and content-sensitive aggregates [44].

3.3. Classification of aggregation schemes

There are many aggregation protocols that can be used for the decentralized monitoring of an aggregated information. In the literature, they are often classified in two prevailing categories: gossip-based protocols and tree-based protocols. Nonetheless, considering only these categories hides many features and differences. Thus, [18,20,23] proposed to consider both network structure (tree-based, gossip-based and hybrid protocols) and propagation technique (reactive and proactive). In this paper, we propose to refine this classification by introducing two other criteria enabling an exhaustive featuring of aggregation schemes, namely neighbourhood information and network view. Consequently, in the following, we consider these four classification criteria all together (network structure, propagation technique, neighbourhood information and network view) as the ground of our taxonomy.

1. **Network structure**: According to the degree of network structure, aggregation protocols can be classified into three categories: tree-based, gossip-based and hybrid protocols. In tree-based techniques, nodes are organized into a tree , and computation is often performed hierarchically in a bottom-up fashion [12,45,46]. Gossip-based aggregation protocols do not require a particular structure, apart from being a connected network. A node contacts one or more of its neighbours usually chosen randomly in each round and exchanges information with them [28,47]. Finally, hybrid protocols combine a gossip-based communication mechanism together with a tree-based structure for aggregation purposes. If several combining solutions exist, hybrid schemes can be

Table 1. Classification of aggregate functions according to the study of Madden *et al*. [42]

	Max, Min	*Count, Sum*	*Average*	*Median*	*Count Distinct*	*Hist*
Duplicate sensitive	×	√	√	√	×	√
Exemplary versus Summary	*E*	*S*	*S*	*E*	*S*	*S*
Monotony	√	√	×	×	√	×
Partial state	Distributive	Distributive	Algebraic	Holistic	Unique	Content sensitive

DOI: 10.1002/nem

exemplified by a tree-based aggregation structure in which each element stands for a cluster of nodes aggregating information in a gossip way [40].

2. **Propagation technique**: It stands for the policy employed to trigger aggregation that leads to the monitoring of given data. There are two propagation techniques: reactive and proactive policies. Nodes use a reactive approach when they compute an aggregation function after having received an explicit request from an AM. It corresponds to the explicit and potentially single polling of an aggregate. By contrast, nodes use the proactive approach to compute aggregation functions if this computation is not triggered by an explicit AM request. That case stands for a continuous monitoring policy. For instance, a proactive scheme can compute an aggregate at each time interval or when changes occur within the network [48,49].
3. **Network view**: It refers to the set of nodes that are considered for the computing of an aggregate. An aggregate could stand for information computed from all the nodes that take part in a managed system and thus provide a global view (GV) or be restricted to only a subset, often determined through a topological criterion, and thus build an SV [8]. The topological distance, given by a number of hops from the AM that requires an aggregate, is the most acknowledged metric for setting the depth of the view. One can notice that aggregation schemes providing a global view do not require the involved nodes to have a full knowledge of the others in the managed domain. For instance, in a basic tree scheme, intermediate nodes only know their children and parent and thus compute a partial aggregate that will eventually lead to a global aggregate at the root. The latter will then be disseminated to all AM, thus providing them with a global knowledge. Concerning the SV, an aggregation protocol is considered as a situated one when it uses an explicit parameter that can be adjusted in order to have a view about some nodes or all the network nodes.
4. **Neighbourhood information**: It determines if communication between nodes is based on a blind or informed policy. In blind communication, nodes select neighbours to exchange information uniformly at random or simply broadcast data to all of them. They do not use any additional knowledge to control their communication policy. By contrast, informed communication methods use heuristics for node selection. In this case, a node explores or selects a subset of its neighbours according to a specific knowledge or a non-uniform probabilistic distribution [49].

4. A SURVEY OF EXISTING SCHEMES FOR DECENTRALIZED AGGREGATED INFORMATION MONITORING

In this section, we provide a brief overview of the main aggregation schemes that have been designed over the last few years. If some of these schemes can be used in standard management frameworks in which control operations do not rely on an autonomic approach, we have considered them because they could even be candidates for monitoring purposes in autonomic management systems. Surveyed schemes are synthesized in Table 2, which features them according to the four criteria identified earlier. Also, considering the two classification criteria related to the network view and structure, in this section, we provide a brief overview of their principle and operation.

4.1. Global view

In global aggregation schemes, the aggregation involves all nodes in the managed system. According to the global data structure, these schemes are either tree-based or gossip-based protocols.

4.1.1. Tree-based aggregation schemes

Tree-based protocols use a hierarchical structure for computing aggregates in a bottom-up manner as depicted in Figure 1. They allow exact values to be computed on the root under a no-failure scenario, contrary to gossip-based algorithms where computation of exact aggregates depends on the convergence time of the algorithm. They also induce a lower overhead by optimizing communication between

DOI: 10.1002/nem

Table 2. Taxonomy of aggregation protocols

Aggregation approach	Network structure	Propagation policy	Network view	Neighbourhood information
Adam2 [50]	Gossip	Reactive	Global	Blind
A-GAP [51]	Tree	Proactive	Global	Informed
AllReport [12]	Gossip	Reactive	Situated	Blind
Astrolabe [40]	Hybrid	Proactive	Global	Blind
Binzenhofer *et al.* [52]	Hybrid	Reactive	Situated	Informed
DECA [53]	Hybrid	Proactive	Situated	Blind
DRINA [54]	Hybrid	Proactive	Global	Informed
Extrema Propagation [55]	Gossip	Proactive	Situated	Blind
GAP [45]	Tree	Proactive	Global	Blind
G-GAP [13]	Gossip	Proactive	Global	Blind
GRASS [56]	Hybrid	Proactive	Global	Informed
Haridasan *et al.* [57]	Gossip	Proactive	Global	Blind
LBA [58]	Tree	Reactive	Situated	Informed
LPS [59]	Gossip	Proactive	Situated	Blind
M-GAP [60]	Tree	Proactive	Situated	Blind
MultipleTree [12]	Hybrid	Reactive	Global	Blind
PM-GAP [60]	Tree	Reactive	Situated	Blind
Propagate2All [12]	Gossip	Reactive	Situated	Blind
RandomizedReport [12]	Gossip	Reactive	Situated	Blind
Randomized Gossip [61]	Gossip	Proactive	Global	Blind
SDIMS [62]	Tree	Reactive	Global	Informed
Sen [63]	Gossip	Proactive	Global	Informed
SG-GAP [64]	Gossip	Proactive	Global	Blind
SingleTree [12]	Tree	Reactive	Global	Blind
Spatial gossip [36]	Gossip	Proactive	Global	Informed
TCA-GAP [65]	Tree	Proactive	Global	Informed
Uniform gossip [28]	Gossip	Proactive	Global	Blind
Willow [66]	Tree	Proactive	Situated	Informed

A-GAP, Adaptive Generic Aggregation Protocol; DECA, DECentralized Aggregation; DRINA, Data Routing for In-Network Aggregation; GAP, Generic Aggregation Protocol; G-GAP, Gossip-based Generic Aggregation Protocol; GRASS, Grid-based Routing and Aggregator Selection Scheme; LBA, Lifetime Balanced data Aggregation; LPS, Local Push-Sum protocol; M-GAP, Multi-Generic Aggregation Protocol; PM-GAP, Pull-based Multi-Generic Aggregation Protocol; SDIMS, Scalable Distributed Information Management System; SG-GAP, Synchronous Gossip-based Generic Aggregation Protocol; TCA-GAP, Threshold-Crossing Alert Generic Aggregation Protocol.

nodes [57]. Nonetheless, there is generally a unique path from each node to the root in such protocols. Thus, a failure of one node will cause the failure of the entire subtree below the failing one. Accordingly, recovery mechanisms are required in a tree when an error occurs. The following is an overview of standard aggregation protocols based on trees.

SingleTree [12] *(tree, reactive, global and blind)*: SingleTree is a generic aggregation protocol very similar to the Echo algorithm [67]. It is based on a spanning tree constructed on the network. To this end, a querying nodes neighbours. The query is flooded over the network, and each node becomes the child of the node that first transmitted the query to it. Once the spanning tree is built, the aggregate is computed in a bottom-up fashion, each node waiting until it receives an answer from all its active children to aggregate the answers and to send the result to its own parent. Thus, SingleTree is a spanning tree-based protocol (tree) where the nodes react to an initial query (reactive) flooded throughout the whole network (global) and hold no specific information on their neighbourhood (blind).

GAP [45] *(tree, proactive, global and blind)*: GAP is dedicated to the continuous estimation of global aggregates that ground many existing aggregation protocols. To this end, GAP builds and maintains a breadth-first search (BFS) spanning tree over the whole network and uses it to

DOI: 10.1002/nem

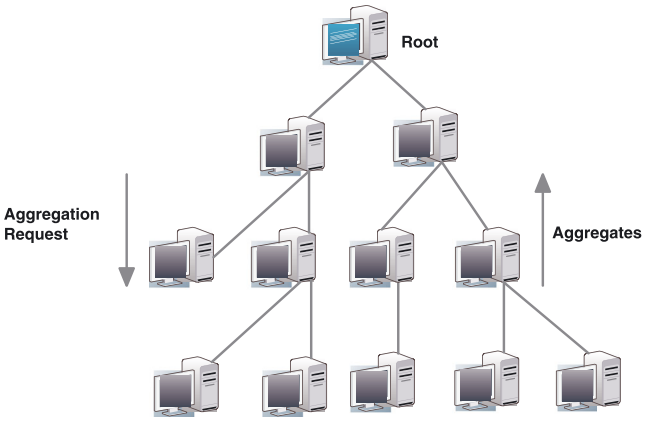


Figure 1. Tree-based aggregation

incrementally and continuously compute and propagate local aggregates back to the root, as in SingleTree. However, each node maintains a table that contains information on itself and on each living neighbour. This table is updated when a failure is detected, or when a new neighbour is discovered. This knowledge is not used to control their communication of aggregates, but to compute complex aggregation functions. Thus, GAP is a spanning tree-based protocol (tree) over the whole network (global) where the nodes continuously propagate aggregates to the root (proactive) and do not use specific information for their communication policy (blind).

Adaptive GAP [51] *(A-GAP; tree, proactive, global and informed)*: A-GAP is an aggregation protocol for the continuous computation of aggregates in a way to ensure scalability and robustness [68]. To this end, A-GAP uses GAP to build and maintain a BFS spanning tree over the whole network. Then, the local values residing at nodes are incrementally aggregated along the tree. However, when the partial aggregate of a node changes, it sends an update to its parent if the difference between the value reported in its last update and the current value exceeds a local filter. Thus, A-GAP controls the overhead by filtering updates that are sent from nodes to the management station passing by the root. The filters periodically adapt themselves to the dynamics of the monitored variables and the network environment. Thus, A-GAP is a spanning tree-based protocol (tree) over the whole network (global) where the nodes continuously propagate aggregates to the root (proactive) and use dynamic filters to control their communication policy (informed).

Threshold-Crossing Alert GAP [65] *(TCA-GAP; tree, proactive, global and informed)*: TCA-GAP is an aggregation protocol for the decentralized detection of threshold crossings. To this end, TCA-GAP uses a spanning tree built with GAP. Then, the root raises an alarm whenever a management variable exceeds an upper threshold and clears this alert when the variable decreases below a lower threshold. TCA-GAP tries to maximize the number of passive nodes in a system through dynamic reallocation of local thresholds. A node is switched to a passive state when its contribution is not needed (e.g. when a node finishes executing the GAP protocol to build a BFS spanning tree, it leaves the active state to enter a passive state). When detecting a threshold crossing, the node enters an active state. Thus, TCA-GAP is a spanning tree-based protocol (tree) over the whole network (global) where the nodes continuously propagate aggregates to the root (proactive) and use local thresholds to control their communication policy (informed).

Scalable Distributed Information Management System [62] *(SDIMS; tree, proactive, global and informed)*: SDIMS is a P2P-based system for aggregating management information. It is based on the Astrolabe system [40] and aims to improve its robustness and scalability when faced with a large number of nodes and a large number of data attributes. To this end, SDIMS relies on Astrolabe for constructing a hierarchy of nodes and uses a distributed hash table (DHT) for exposing a set of multiple aggregation tree overlays on top of managed elements. Thus, SDIMS is a spanning

DOI: 10.1002/nem

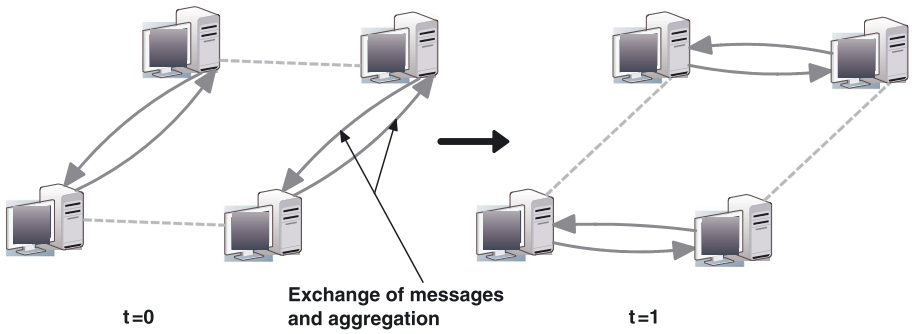


Figure 2. Gossip-based aggregation

DHT-based protocol (tree) over the whole network (global) where the nodes continuously propagate aggregates to the root (proactive) and use the properties of the DHT to control and optimize their communication policy (informed).

4.1.2. Gossip-based aggregation schemes

Gossiping is a technique that allows the spreading of information over a network. In basic gossip algorithms [61], a random pair of neighbouring nodes is chosen each round to exchange information and to update their local values, as shown in Figure 2.

According to [33], gossip-based protocols can be divided with respect to node selection into two categories: uniform gossip and standard gossip protocols. In uniform gossip, each node chooses to exchange information with a uniformly chosen node at each step [28]. In standard gossip, a node chooses, according to a non-uniform probabilistic distribution, one of its neighbours [36,69].

Apart from network monitoring, because of their characteristics (e.g. decentralization, scaling and robustness), gossip-based protocols have been considered in a wide range of contexts such as peer sampling, ad hoc routing, reliable multicast, database replication, failure detection and data aggregation [70]. The large number of messages transferred from node to node in a gossip-like way over the network is the main cause of a high overhead [14,28].

- *Uniform gossip* [28] *(gossip, proactive, global and blind)*: Uniform gossip is a basic gossip protocol introduced by Kempe *et al.* [28] and Jelasity *et al.* [48] that can be implemented on any kind of network. This protocol is proposed to support simple aggregate functions like min, max, average and count. To this end, each node periodically selects one of its direct neighbours uniformly at random in order to exchange their information. When a node receives a new aggregate value from a neighbour, it computes a new aggregate and updates its local state. During the aggregation process, the distance between distributed partial aggregates progressively decreases, and nodes converge to a unique global aggregate. Thus, Uniform Gossip is a gossip-based protocol (gossip) over the whole network (global) where the nodes continuously propagate local aggregates to their neighbours (proactive) chosen uniformly at random (blind). Let us notice that Wuhib *et al.* [13] propose an extension of this protocol to provide accurate estimates in the event of node failures of different types. Each node propagates a summary of its computation in the network, and in case of failure, the other nodes can remove the former from their own computation.

  Spatial gossip [36] *(gossip, proactive, global and informed)*: Spatial gossip is a gossip protocol dedicated to compute simple aggregate functions like min, max, average and count that can be deployed over a network inside a metric space. To this end, each node is defined by coordinates in this metric space, and then a node i selects one of its neighbours j with a probability proportional to $1/d^{\rho}$, where d represents the distance between i and j and ρ is a constant parameter. Therefore, a node closer to the sender has more chances to be selected as a receiver. Thus, spatial gossip is a gossip-based protocol (gossip) over the whole network (global) where the nodes continuously propagate local aggregates to their neighbours (proactive) chosen according to their

DOI: 10.1002/nem

distance (informed). Let us notice that spatial gossip is a generalization of the geographic gossip algorithm [71] where $\rho = 0$.

Haridasan et al. protocol [57]: This protocol is a uniform gossip protocol that allows each node to estimate the distribution of values held by other nodes. To this end, each node i maintains a fixed-size array of k numerical values x_i, which measure some variable of interest of the system. Once a node i receives the array of values from a node j, node i has $2k$ values that are merged into an array of size k by using data synopsis techniques [72]. Thus, this protocol is a uniform gossip protocol (gossip and blind) over the whole network (global) where all nodes continuously estimate a set of aggregates (proactive).

Adam2 [50] *(gossip, reactive, global and blind)*: Adam2 is an aggregation protocol that computes the cumulative distribution function over an attribute in a network. To this end, first, the protocol initializes a set of thresholds and runs two instances of basic gossip protocols: an averaging protocol over the attribute and a counting protocol to estimate the size of the network. When the averaged attribute reaches a threshold, the current number of nodes counted by the protocol is associated to the latter. Finally, the set of all thresholds and the associated values allow Adam2 to compute the cumulative distribution function. Thus, Adam2 uses several basic gossip protocols (gossip and blind) over the whole network (global) where the nodes aggregate the data according to a set of thresholds chosen by an AM (reactive).

Sen [63] *(gossip, proactive, global and informed)*: This protocol is a standard gossip protocol dedicated to limiting the communication within the network. To this end, it is initialized with a precision parameter. Every node maintains an estimate of the current global aggregated value. Each time a node receives new information, it computes a new global estimate, and then it gossips this new estimate to its neighbours if the difference between the old and new estimates exceeds the precision parameter. When a node receives an estimate from a neighbour, it only aggregates this value if the difference between its own estimate and the one that has been sent does not exceed the precision. Thus, this is a gossip-based protocol (gossip) over the whole network (global) where the nodes continuously propagate local aggregates to their neighbours (proactive) chosen with respect to a precision parameter (informed).

4.1.3. Hybrid aggregation schemes

In order to combine the benefits of both gossip-based and tree-based protocols, some protocols propose a hybrid approach. In such approaches, the aggregation protocol uses a gossip communication combined with a tree structure.

MultipleTree [12] *(hybrid, reactive, global and blind)*: MultipleTree is an enhancement of SingleTree that aims to provide robustness to the aggregation process. To this end, it creates k (typically two or three) independent spanning trees rooted at a querying node. A request is flooded from the querying node to the whole network in order to give the nodes a level (the length of the shortest path from a node to the root). Then, each node picks uniformly at random k parents from its neighbours that have a lower level than it has. Finally, the aggregate function is calculated in a bottom-up fashion as in SingleTree. Thus, MultipleTree is a spanning tree-based protocol where the nodes have several parents (hybrid), react to an initial query (reactive) flooded though the whole network (global) and hold no specific information on their neighbourhood (blind).

Grid-based Routing and Aggregator Selection Scheme [56] *(GRASS; hybrid, proactive, global and informed)*: GRASS is an aggregation protocol dedicated to wireless sensor networks. It aims to optimize the energy consumption within the whole network while maintaining a GV on a given aggregate. To this end, the protocol partitions the network into several groups of nodes that may overlap. Each group is coordinated by a special node, called master aggregator. A local aggregation is carried out inside each group via a gossip protocol. Then, these local values are propagated along a tree of master aggregators where the root is the sink. This tree is computed offline thanks to a genetic algorithm in order to optimize the energy consumption of the whole network. Thus, GRASS combines tree-based and local gossiping communications (hybrid) where

DOI: 10.1002/nem

a sink continuously monitors a given aggregate (proactive) on the whole network (global) and uses offline optimization to control its communication policy (informed).

Astrolabe [40] *(hybrid, proactive, global and blind)*: Astrolabe is a distributed information management scheme that monitors and reports to users the dynamically changing state of a set of resources. To this end, it uses gossip to construct an overlay tree for computing aggregates. The tree is only an abstraction of a hierarchical relational database, constructed using a P2P protocol by agents running at each node. Each node keeps a local copy of the data for its own region and summary data for the region above it on the path to the root of this hierarchy. When changes occur, they are spread throughout the system using a gossip communication. Thus, Astrolabe uses both tree-based and gossip-based communications (hybrid) where a root node continuously monitors a given aggregate (proactive) on the whole network (global) and does not use specific information for its communication policy (blind).

4.2. *Situated view*

Situated approaches propose to reduce the aggregation cost by limiting the knowledge of a node to a bounded neighbourhood. Thus, in opposition to the global schemes where global aggregates are computed over the entire network, each node computes a partial aggregate representing its partial view by collecting data from its neighbourhood or a subset of the network nodes. The size of this view is defined by a number of nodes or a number of hops, as shown in Figure 3.

4.2.1. *Tree-based aggregation schemes*

Multi-GAP [60] *(M-GAP; tree, proactive, situated and blind) and pull-based multi-GAP (PM-GAP; tree, reactive, situated and blind)*: M-GAP is an extension of GAP, where the partial aggregates are available at all nodes. To this end, a node first computes a partial estimate of the aggregate over the partial aggregates received from its children and its parent in the BFS spanning tree. In this case, an update vector contains not only a single partial aggregate as in GAP but a list of partial aggregates too. Each node that receives an update vector uses that list to update its local neighbourhood table. Thus, M-GAP is a spanning tree-based protocol (tree) where the nodes continuously monitor (proactive) the set of partial aggregates computed by their neighbours (situated) without using specific information in their communication policy (blind). Let us notice that there is an extension of M-GAP named PM-GAP [60] where each node pulls the update vectors from their neighbours by punctual queries (reactive) instead of continuous monitoring.

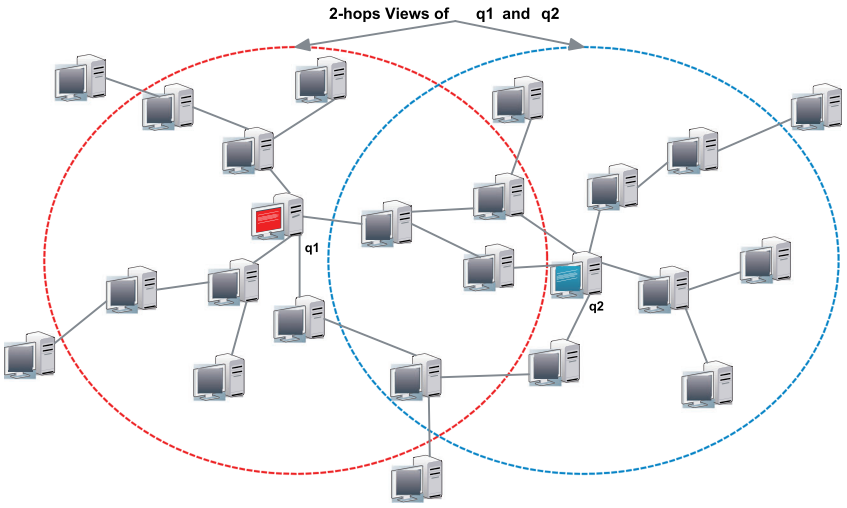


Figure 3. Situated aggregation

DOI: 10.1002/nem

Lifetime Balanced data Aggregation [58] *(LBA; tree, reactive, situated and informed)*: LBA is an aggregation protocol designed such that at least p per cent of local data should be delivered to a sink within time D after the data have been generated. To this end, the sink builds a spanning tree where child nodes, while transmitting data to their parent, report their current lifetime, forwarding aggregation delay, self-aggregation delay, data output rate and data input rate. Based on this information, the parent node uses a strategy (analytically defined offline) to determine if it will immediately increase (or decrease) its aggregation delay and if it needs to ask its children to decrease (or increase) their aggregation delays. Thus, LBA is a spanning tree-based protocol (tree) where an explicit subset of the network (situated) answers an aggregation request (reactive) and uses specific information to optimize its communication policy (informed).

4.2.2. Gossip-based aggregation schemes

Propagate2All [12] *(gossip, reactive, situated and blind)*: Propagate2All is a gossip protocol dedicated to compute an aggregate function on a subset of the network. To this end, a querying node floods a query message with a parameter $\hat{D}$ that denotes the upper bound to which the network must be known. Each node that receives the query becomes active and computes its local aggregate through a uniform gossip protocol. Each node stops $2\hat{D}\Delta$ time after it became active, where Δ is the maximum delay between any pair of nodes. Thus, Propagate2All uses a uniform gossip protocol (gossip and blind) where a querying node requests (reactive) a partial view on a subset of the network (situated).

AllReport and RandomizedReport [12] *(gossip, reactive, situated and blind)*: AllReport is an aggregation protocol used to sample live nodes in an unknown network. To this end, a querying node q sets a timer, equal to a given maximum round-trip time 2Δ, and broadcasts a query over the network. Each node that satisfies the query sends the aggregation data directly to q. After receiving the data, the querying node computes the aggregate and resets the timer to reach other nodes. RandomizedReport is an extension of AllReport that aims to reduce the number of messages sent to the querying node. To this end, this node initiates a broadcast message that contains a sampling parameter p, and it sets its timer to expire after $\hat{D}\Delta$ to reach all the nodes. A receiving node replies to the querying node with probability p and passes the message to its own neighbours. Thus, both protocols use a uniform gossip protocol (gossip and blind) where a querying node requests (reactive) a partial view on a subset of the network (situated): a temporal sampling for AllReport and a probabilistic sampling for RandomizedReport.

Local Push-Sum protocol [59] *(LPS; gossip, proactive, situated and blind)*: LPS is an averaging uniform gossip protocol that monitors only a partial subset of the network. To this end, the nodes share their values with their direct neighbours only and calculate the aggregates characterizing the sole spatial region they belong to. Thus, LPS is a gossip-based protocol (gossip) over a subset of the network (situated) where the nodes continuously propagate local aggregates to their neighbours (proactive) chosen uniformly at random (blind). Let us notice that Gouvas *et al.* [73] have proposed an informed variant of LPS in order to converge faster by introducing a precision threshold.

Extrema Propagation [55] *(gossip, proactive, situated and blind)*: Extrema Propagation is a protocol dedicated to approximate the number of nodes in a given network at a chosen precision, and in a short number of message exchanges. To this end, each node generates a vector of random real numbers according to a known probability distribution (e.g. Gaussian or exponential) and aggregates all the vectors through a minimum function with a uniform gossip protocol. In each round, every node sends a message containing this vector to its neighbours, collects the corresponding messages from them and computes the pointwise minimum of those vectors. The resulting vector corresponds to a new distribution that can be used to infer the number of nodes by a maximum likelihood estimator. Thus, Extrema Propagation is a uniform gossip protocol (gossip and blind) where nodes continuously (proactive) estimate the size of the network at a given precision (situated).

DOI: 10.1002/nem

4.2.3. Hybrid aggregation schemes

DECentralized Aggregation [53] *(DECA; hybrid, proactive, situated and blind)*: DECA is a monitoring protocol for structured networks based on DHTs. To this end, nodes are organized in a hierarchy of self-contained clusters: topologically close nodes are organized into clusters of fixed cardinality, and topologically close clusters are organized into super-clusters. Each node periodically selects a random subset of nodes from its cluster and computes a partial aggregate that is finally sent to a delegated node from the super-cluster. Then, an aggregate is computed in a bottom-up way until the top of the hierarchy is reached. Thus, DECA uses both tree-based structure and intra-cluster uniform gossiping communication (hybrid and blind) where each node continuously monitors an aggregate for which the view depends on the node's place in the hierarchy of self-contained clusters (situated).

Data Routing for In-Network Aggregation [54] *(DRINA; hybrid, proactive, situated and informed)*: DRINA is an aggregation scheme for wireless sensor networks. To this end, a tree is built from a sink node that acts as the root. Then, when a new event is sensed by a node, this information is shared with its whole neighbourhood. Finally, the node that is closest to the sink node in the tree is elected among all nodes that sensed the same event. This node is in charge of aggregating the local information and then routes the data to the sink through the tree. Thus, DRINA is a tree-based protocol over cluster of nodes (hybrid) where each node continuously monitors (proactive) a given neighbourhood (situated) and elects online the best node to route the data to the sink (informed).

Binzenhofer et al. protocol [52] *(hybrid, reactive, situated and informed)*: This protocol is a snapshot algorithm for Chord [74], a P2P system based on DHTs, to monitor a distinct division of a network. To this end, the algorithm is called by an arbitrary peer to divide recursively the overlay into contiguous subparts of a predefined size. Then, the first peer in the region creates a token, adds its local aggregate and passes it to its immediate successor, which updates and passes recursively the token until reaching the last peer in the subpart, which sends back the token to a central collecting point. Thus, this protocol first uses a tree-based DHT and then directs neighbourhood communications (hybrid) to allow an AM (reactive) to monitor distinct subsets of a given network (situated) by using the properties of the DHT (informed).

5. EVALUATION STUDY

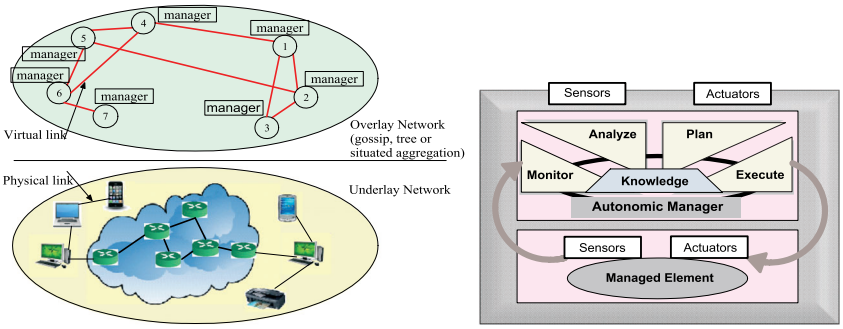
In order to highlight the different operational contexts in which an aggregation scheme outperforms the others for autonomic management, we have implemented, for each previous aggregation category, an abstracted version of a scheme that acts as a consensual representant of approaches proposed in the state of the art. Thus, we have implemented two global schemes (i.e. tree and gossip) and one situated scheme with different sizes of the node views. In this section, we present the management context in which our study is placed, the algorithms we used, the factors that impact the performance of these schemes and the way we can quantify these impacts through the definition of performance metrics.

5.1. General context

The general approach that we followed in order to establish the operational context in which aggregation schemes have been evaluated is guided by (i) the need to make results as generic as possible so that they could be even partially adapted in further contexts by researchers or autonomic management system designers and (ii) the need to stress the considered schemes to highlight their intrinsic limits. Obviously, such an approach also presents its own limits given that all possible contexts cannot be handled in a single study, making the reusability of results for very specific purposes limited.

From a management perspective, as depicted in Figure 4(a), we consider a full autonomic management system in which every managed element integrates an AM that operates with the others in a purely decentralized way. The information it processes in local control loops (Figure 4(b)) is considered as a stream of data, leading to a continuous monitoring activity. This context covers numerous

DOI: 10.1002/nem



(a) (b)

Figure 4. (a) Our decentralized management use case. (b) The particular case of autonomic management [2]

Table 3. Notations

Notation	Signification
N	Number of nodes in the managed domain
X_i	Local aggregate of node i
X_{raw_i}	Raw (non-aggregated) value of node i
X_{thr}	Threshold value leading to a control action in the managed domain
w_i	Weight of X_i
state_i	Local state of node i
h	Maximum number of hops
$\mathbb{D}_i$	Set of direct neighbours of node i
$\mathbb{L}_i$	Set of nodes in the situated view of node i

management use cases such as the monitoring of the average traffic related to a specific service (such as Hypertext Transfer Protocol as considered in the GAP paper and its followers) or the amount of shared data in a P2P file-sharing network. The aggregation functions we consider are only mathematical ones because they represent the core of the monitoring activity in network and software service management. We do not address complex aggregation functions such as compression or correlation. In order to stress the protocols, we consider that managed elements are end hosts that are disseminated over the whole Internet and thus connected through end-to-end communications. Thus, on the service plane, it appears as a multi-domain management case where all managed elements are gathered in a single management domain that forms an overlay. Managed elements hosted by terminals exhibit a dynamic presence that leads to churn in the managed system. However, in order to keep the generic aspect of our study, although they could stand for mobile devices connected by wireless networks, we do not consider any energy aspect.

5.2. *Implemented aggregation schemes*

According to the classification of aggregation schemes presented in Section 4, three main aggregation categories emerge: tree-based, gossip-based and situated approaches. Moreover, when looking at these schemes, it appears that their performance limits are mainly due to the intrinsic kind of approach they rely on. As a consequence, most proposals are enhanced with additional mechanisms that (i) are designed for a specific aggregation function, (ii) use specific knowledge on the monitored data. However, such enhancements do not allow a clear identification of the context in which they are most suited. This is why we have chosen to implement one representative scheme from each of these aggregation categories. We only implemented the basic operations for each scheme, which are often based on the proposal of a legacy approach that has led to several subsequent improvements, so that a clear contextual area could be highlighted.

DOI: 10.1002/nem

Algorithm 1 Push tree-based aggregation algorithm executed on node i

(a) Active thread

1: p←getParent()
2: **if** (p!=null) **then**
3: send $(< X_i, 1 >, p)$
4: **end if**
5: receive(X_j, j)
6: $state_i$←update(X_j)

(b) Passive thread

1: **loop**
2: receive $(< X_j, w_j >, j)$
3: $state_i$←update(X_j, w_j)
4: p←getParent()
5: **if** (p!=null) **then**
6: send $(< X_i, w_i >, p)$
7: **else**
8: wait until receive all states (timer)
9: diffuse (X_i)
10: **end if**
11: **end loop**

Algorithm 2 Push-pull gossip-based algorithm executed on node i

(a) Active thread

1: **loop**
2: j←getNeighbors(1)
3: send (X_i, j)
4: receive(X_j, j)
5: $state_i$←update(X_j)
6: wait(round duration)
7: **end loop**

(b) Passive thread

1: **loop**
2: receive(X_j, j)
3: send (X_i, j)
4: $state_i$←update(X_j)
5: **end loop**

All the notations used in this part are summarized in Table 3.

5.2.1. Tree

In this category, we implemented a proactive push tree-based aggregation scheme that acts as a basic abstraction of the GAP [45] and SDIMS [62] schemes. Our scheme consists in both collecting and computing aggregates (in a bottom-up way) but also diffusing them (in a top-down way) to a managed community. Thus, after convergence, each node of the tree will have the same global aggregate, contrary to classical tree-based schemes where only the root maintains a global aggregate. We assume that the diffusion part is performed through the use of any existing approach that could be, for example, multicast-based in case of a single-domain management or an application-layer publish/subscribe mechanism in case of multi-domain management [75].

As shown in Algorithm 1, which describes the main operations of the developed tree-based scheme, one active thread and one passive thread are executed on each node. The first one initiates the aggregation process by sending the value of each node to its parent. It is executed once at each iteration of the collecting information process. The second thread waits for messages sent by an initiator (i.e. active thread) to process them. Initially, each node i obtains its current parent with the help of the getParent() method. If it has at least one parent, it sends it a couple $\langle X_i, 1\rangle$ of the form $\langle$aggregate, weight$\rangle$ as given in lines a.1–a.4. A node i that receives a message from a child node j computes a new partial aggregate according to a given aggregate function, updates its local state with the new values and then forwards them to its parent p as shown in lines b.2–b.6. If node i is the root (i.e. p = null), then it waits a certain time until it receives all its children's aggregates, and it incrementally diffuses the computed global aggregate X_i over a publish–subscribe scheme on all the interested nodes of the tree in a top-down way as given in lines b.7–b.10. Thus, each node that receives X_i from the root updates its partial aggregate with the global one as shown in lines a.5 and a.6.

DOI: 10.1002/nem

Algorithm 3 Pull situated view algorithm on node i

(a) Active thread

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

(b) Passive thread

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

5.2.2. Gossip

The aggregation scheme developed here is the basic push–pull gossiping scheme [48] with symmetric information exchanges where nodes both send and receive their estimates.

As illustrated in Algorithm 2, with the help of the getNeighbors(1) method that selects uniformly at random one node j from a list of direct neighbours $\mathbb{D}_i$, each node i sends to j a message containing its partial aggregate X_i and waits for a response from the remote node j as shown in lines a.2 and a.3. When i receives a response (X_j, j) from j, it updates its local state through the update() method that computes a new partial aggregate according to the selected aggregate function as given in lines a.4 and a.5. The node i waits for a duration *round duration* and then repeats the same process (line a.6) at each round. Similarly, when the passive thread of node i receives an exchange request message, it replies with its local aggregate X_i, and then it updates its local state by computing a new aggregate through the received one, as shown in lines b.2–b.4.

5.2.3. Situated view

We have implemented a typical situated scheme inspired from a membership protocol named Hyper Partial View (HyParView) [76] proposed to maintain partial views for gossip protocols. In HyParView, each node maintains two situated views: an active view maintained with a reactive strategy that responds to events in the system and a passive view maintained proactively with periodic updates of management information. Aggregation is only performed when a node receives information from its neighbours and decides to update its own information or not. In our situated scheme, each node maintains one partial view of a part of network nodes, bounded by a maximum number of hops. A node i can thus obtain an aggregate of its view by collecting management information from its h-hops neighbours.

As shown in Algorithm 3, the requesting node i obtains the updated list $\mathbb{D}_i$ of all its direct neighbours through the getNeighbors(all) method and sends them a query message containing the maximum number of hops h as given in lines a.1 and a.2. Each node i that receives an aggregation request message (line b.2) verifies if it did not previously answer the same request coming from j in the same aggregation cycle. If so, i answers by sending a response (X_{raw_i}, j) directly to the requesting node j (line b.3). Then, node i decrements the number of hops contained in the received message (line b.4). If the maximum number of hops is not reached, then node i forwards the request to all its direct neighbours as given in lines b.5–b.8. When a requesting node i receives an answer (X_{raw_j}, j) from a neighbour j, it computes a new partial aggregate and updates its own state as shown in lines a.3 and a.4.

5.3. Impacting factors

The factors that impact the performance and the cost of decentralized aggregation schemes have been identified in the literature, as criteria for their evaluation. Among them, we selected those that could be quantified in order to lead to different design choices in the context of an autonomic management framework. These are the network size, the network dynamics and the aggregated information

DOI: 10.1002/nem

dynamics. Two other factors, namely the network topology or the kind of aggregation function, may be considered. However, both of them can only be qualitatively rather than quantitatively defined (e.g. ring, scale free or regular for network topology and count, count distinct or histogram for aggregation functions). Consequently, we only consider quantitative factors, while qualitative factors are left for future work.

In the following, for each of the four factors we chose, we define it, show how it impacts the aggregation scheme operation and also provide numerical values that serve as the reference range for our evaluation framework. One can note that these values are chosen to cover use cases found in the literature, but some of them are restricted because of pragmatic limits of the evaluation framework (e.g. number of nodes).

5.3.1. Network size

As distributed algorithms, decentralized aggregation protocols are impacted by the number of elements that take part in their operation, which in our case stands for the number of AMs that run the aggregation algorithm. Indeed, analytic studies of their complexity have shown that their cost (e.g. number of exchanged messages) as well as their performance (e.g. convergence time) is dependent on the number of nodes. However, if their different complexities do not induce a significant difference at small scales, the latter could become an important criterion at larger ones for the adoption of a scheme against other ones. In the following, we consider a number of AMs in the managed domain that varies between 2 and 1000 nodes. This upper bound is mainly due to the computation limits of the simulation framework we used. However, one can note that our limit of 1000 nodes is comparable with the upper limits used in the evaluation of the presently evaluated schemes by the authors in their original papers. For instance, the set of protocols belonging to the GAP family was evaluated from about 500 [45] up to 10 000 nodes [77] as an upper limit, while SDIMS [62] was evaluated with 4096 nodes. The sole approach that was evaluated with a significantly larger order of magnitude is the gossip of Jelasity *et al.* [48], which was evaluated in a 100 000-node network.

5.3.2. Network dynamics

The network dynamics, or churn, consists in the nodes' arrivals in the system and their departures that can be planned or due to a sudden failure. This parameter allows us to investigate the redundancy and resilience of a given protocol to communication failures. Based on the analysis of the traces on different networks like mobile ad hoc networks and P2P ones, many prior studies assumed a Poisson process of parameter λ for the node arrivals and an exponential distribution of parameter μ for the departures [78–80]. Thus, we adopt the same models in our experiments, and we choose the values of arrival and departure rates so that they cover values used in existing networks. More specifically, the values of λ and μ are chosen so that the average number of nodes, given by $N = \lambda/\mu$, is equal to 500. Thus, in our study, λ varies in [0.0041; 0.4166], and μ ranges in $[8.33 \times 10^{-6}; 8.33 \times 10^{-4}]$, corresponding respectively to average inter-arrivals ($1/\lambda$) ranging in [2.4; 240] s and to average lifetime duration ($1/\mu$) in the range [20; 2000] min.

5.3.3. Information dynamics

It determines the evolution degree of management information over time. It can be characterized through two criteria: (i) the changing frequency of the values in managed elements, for example, a value X_{raw_i} whose instrumentation brings to the management plane an updated value every second; and (ii) the changing degree of this management information that is defined by the distance between two consecutive values. In order to clearly identify the way this phenomenon impacts the aggregation process but also to provide our framework with a pseudo-realistic model, we model it through a gauge evolving in a linear way. Such information could, for example, stand for the average traffic throughput of a managed domain aggregated by the number of bytes emitted by all the managed elements over a given period. Thus, X_{raw_i} is periodically incremented by α until reaching 100, and it is then decremented by the same α until 0. Through that model, we are able to control the information

DOI: 10.1002/nem

dynamics level. One can note that we do not choose to use the trace of a concrete captured management information because, although it would have brought us more realistic results, it would also have made more complex the isolation of the dynamics impact because the latter would have changed continuously during the execution of a simulation unit. By contrast, through our model, by fixing α, we completely master it and are thus able to isolate its impact on simulation results. Also, by extension, phenomena where some information would change radically are captured through both the highest changing frequencies and values. Numerically, the local values range in [0; 100] for an α that we vary in [0.02; 5].

5.4. Evaluation criteria

With the same methodology we used for the choice of impact factors, we determined metrics and indicators that feature the performance and cost of aggregation schemes through the use of acknowledged criteria in the literature [9,12–14] and also through the objective of helping users of autonomic management in the choice of an approach. This is why we choose convergence time, communication and computation cost, scalability and accuracy as performance metrics, complete with the computation of the impact of accuracy on the decision-making process as a performance indicator.

5.4.1. Convergence time

In the context of the use of decentralized aggregation schemes for autonomic management purposes, we define the convergence time as the necessary time between the initialization of the aggregation process and the time t when all AMs hold the aggregation results. This definition means that it includes both the time for aggregating data and also the one for disseminating it to AMs. Thus, it is consistent with those used in the evaluation of gossip-based aggregation schemes, but it differs from those used for tree-based schemes because the dissemination is performed once the aggregate is computed at the tree root [12]. Thereby, we define it through $T_{\text{conv}} = T_{\text{agg}} - T_{\text{init}}$, where $T_{\text{init}} = t, \forall i \in \{1, \ldots, N\},\ X_i^t = X_{\text{raw}_i}$. In the case of GV, T_{conv} corresponds to the time when all nodes hold the same global aggregate. Thus T_{agg} is expressed by $T_{\text{agg}}(GV) = t, \forall i, j \in \{1, \ldots, N\},\ |X_i^t - X_j^t| < \epsilon$. In the situated scheme, this condition cannot be reached because each node retrieves only the values of its h-hops neighbours. So, we measure the required time for each node to collect information from its neighbours and to calculate a partial aggregate.

5.4.2. Communication cost

The communication cost [12] is the sum of the 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, $C_{\text{com}} = \sum_{i=1}^{N} C_{\text{com}_i}$, where C_{com_i} is the the number of messages sent by node i.

5.4.3. Computation cost

The computation cost [12] 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, $C_{\text{cpt}} = \max(C_{\text{cpt}_i}), i \in \{1, \ldots, N\}$.

5.4.4. Accuracy of estimated aggregates

The accuracy of an aggregation scheme stands for the distance between the actual value of an aggregate and the one estimated by the aggregation scheme. Such a featuring highly depends on the nature of the aggregate function. For example, in the case of the aggregation of an average value, it could be expressed as 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 under a static environment and to show how

DOI: 10.1002/nem

far these values lie from the average value, we compute the variance over all the estimates through $V(X) = \frac{1}{N}\sum_{i=1}^{N}(X_i - \overline{X})^2$.

Under a dynamic environment, the variance cannot be directly used to measure accuracy. This is because even if the variance is very low, the average value can be far from the actual value of the global aggregate that should be obtained at this moment. Thus, in order to evaluate the aggregation accuracy under a dynamic environment, we look at the distribution of estimates over all the network nodes, and we measure the distance between estimated aggregates X_i and the actual global aggregate X that should be computed (i.e. average deviation). The latter is computed through $D_X = (1/N)\sum_{i=1}^{N}|X_i - X|$.

5.4.5. Efficiency of threshold-crossing detection

Measuring the accuracy of an aggregation scheme provides a good indicator of its performance but is not sufficient. Indeed, considering situated schemes, their intrinsic operation will, most of the time, lead to inaccurate estimations of aggregates, while their use in autonomic control frameworks is now acknowledged. Thus, we propose to evaluate the impact of the aggregation accuracy on the control part. As established in the Monitor–Analyse–Plan–Execute [2] control loop (Figure 4(b)) forming the core of AMs, if the main focus of this study is related to the 'Monitor' functional block, we basically feature here the impact of its accuracy on the 'Analyse' one. To that aim, we propose to model this block through a standard decision rule used in network management relying on a threshold value. Such a rule can be expressed through $X_i > X_{\text{thr}} \rightarrow D_1$ and $X_i \leq X_{\text{thr}} \rightarrow D_2$.

For each aggregation cycle, at the end of the process, we compare the resulting decisions on each node with the one that would have been made in the case where it would have obtained the current actual global aggregate. Thus, we count the number of decentralized decisions that do not match with the one that should have been taken, given the actual aggregated value, and we consider them as faulty decisions. One can note that featuring the impact of the monitoring performance on decision-making processes in autonomic management is an entire work that is out of the scope of this study but would be a track for future work.

6. EVALUATION RESULTS

On the basis of the autonomic management context, impact factors and the performance metrics identified in previous section, we now present the evaluation results we have obtained through an exhaustive simulation campaign of the decentralized aggregation schemes we also presented previously. First of all, we introduce the simulation framework we have built. Then, for each performance metric, we analyse the results we obtained. Finally, we provide a synthesis of all these results, leading to abstract guidelines for the choice of a decentralized aggregation approach.

6.1. Testbed and scenarios

The management plane we consider relies on an overlay (Figure 4), and the framework we choose to provide basic functionality is the FreePastry[1] simulator, an open-source Java implementation of the Pastry DHT [81]. We carry out all our experiments with the Euclidean physical network topology model. In this model, hosts are placed randomly and uniformly within a two-dimensional Euclidian space. The distance between two hosts models the latency in the routing. Then, the topology we used for the network overlay, where all the aggregation schemes are executed, is given by Scribe [82], an application-layer publish–subscribe approach to spread the root's aggregates on all the AMs, for the tree-based protocol. The overlay topology we used for the gossip-based and SV protocols is a scale-free network [83]. Let us notice that, for each kind of these protocols, we choose a beneficial topology in order to compare their best performances. Thus, the performance of each kind of protocol should

[1] *http://freepastry.org*

DOI: 10.1002/nem

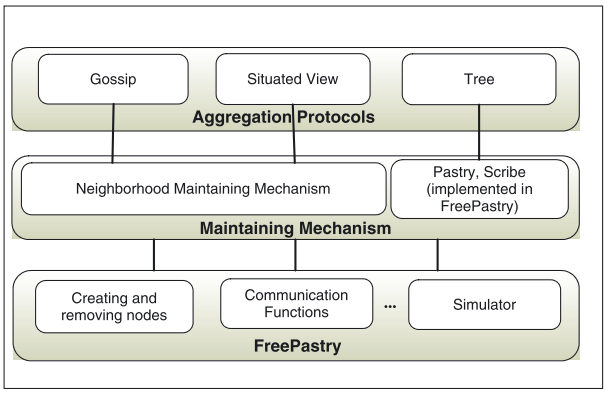


Figure 5. Architecture of the evaluation framework

Table 4. Simulation parameters

Parameter	Value
Aggregate function	Average
Physical network topology model	Euclidean
Overlay topology model	Scribe and scale free
Topology maintaining frequency	200 ms
Values changing frequency	20 s
Tolerated error (ϵ)	0
Neighbourhood degree	8
Gossip round duration	600 ms
Number of nodes (N)	[2; 1000]
Number of hops in SV (h)	[1; 2]
Decision threshold (Thresh)	50

be lowered if they are used on a non-beneficial topology (for instance, on a random network for SV schemes). The architecture of this evaluation framework is illustrated in Figure 5.

According to the set of metrics we evaluated, we consider two different scenarios. Convergence time, communication and computing costs are evaluated through a static scenario in which both the network topology and the aggregated information do not change over the simulation. In this context, each of the developed aggregation schemes computes an average of randomly generated values ranging between 0 and 100. As for accuracy and impact on a threshold detection process, we evaluate them in both the static scenario and a dynamic one in which AMs join and leave the network according to our churn model, while the managed information evolves over time through the previously presented linear gauge model.

Based on the parameters summarized in Table 4, the situated scheme is executed with a view limited to the direct neighbours (SV1) and also with two hops (SV2). The gossip process is executed with rounds of 600 ms that correspond to the maximum time for an information exchange. Let us notice that we fixed the tolerated error ϵ to 0. In terms of implementation, it is not exactly 0, but the floating point precision of the machines we used. We choose this value in order to minimize the trade-off between accuracy and the other parameters. As said previously, the number of nodes we choose seems to be sufficient to obtain scaling results. To give a sufficient statistical significance to these results, each value presented here is an average of the values obtained on 100 executions of the aggregation algorithms. We also provide 95 % confidence intervals on all depicted measurement points.

DOI: 10.1002/nem

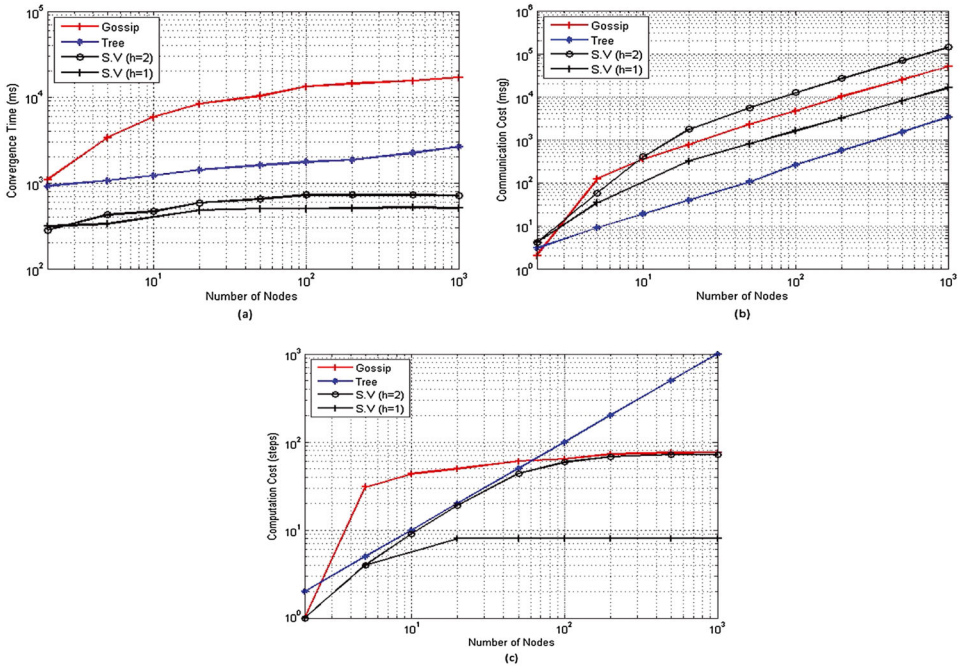


Figure 6. Aggregation cost regarding: (a) convergence time, (b) communication cost and (c) computation cost (log–log scale)

6.2. *Analysis*

For each evaluated metric, we present the result obtained through the simulations, and in order to provide guidelines for the choice of an aggregate scheme for a given operational context, we highlight differences in the order of magnitude between the considered schemes. We also provide comments on the impact of the use of a scheme on the managed system and the management control loop.

6.2.1. *Convergence time*

We observe in Figure 6(a) a large convergence time for the gossip scheme followed by the tree and then a relatively low time for the situated view. Under a network of 1000 nodes, the gossip's convergence time is about six times higher than that of the tree and about 23 times that 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 neighbours and then computes a partial aggregate of the received values to two-hop nodes. This time is lower in SV1 when only direct neighbours are contacted. Thus, in terms of convergence time, the situated scheme scales better and converges more quickly than the global ones, thus enabling autonomous managers to be more reactive to events when relying on this scheme.

It is interesting to notice that the choice of the aggregation scheme can have an impact on the management decision process. Indeed, if the values changing frequency is high, it seems to be better to use an SV scheme, whereas gossip-based protocols seem to be more adapted if the values changing frequency is low. However, an efficient management decision-making is also based on the accuracy of the protocols (as discussed in Section 6.2.5)

6.2.2. *Communication cost*

Figure 6(b) exhibits that for all schemes, the communication cost is proportional to the number of nodes. At worst, for a 1000-node network, the communication cost of SV2 is almost three times higher

DOI: 10.1002/nem

than that of gossip and about 42 times that of the tree. This high communication cost is explained by the fact that the former scheme is based on a broadcast algorithm where each node floods its request message on all its h-hop neighbours. By contrast, SV1 causes more communication overhead than the tree but less than the gossip. In this scheme, a node only exchanges its value with its direct neighbourhood, thus inducing about nine 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 (we recall that the root does not directly communicate with the leaves as the tree is used for all communications). For the gossip scheme, it involves more messages to converge than the tree because it uses a blind communication over multiple rounds. Thus, if the situated scheme seems scalable regarding the local communication cost, at the managed system level, its flooding approach induces a very important overhead. This is why considering a situated scheme for autonomic monitoring of aggregated information has to be carefully studied and the impact of the communication cost of the management traffic over the regular service traffic has to be addressed in order to avoid a too costly management framework and, even more, the occurrence of collapse points in the network because of too important management traffic.

6.2.3. Computation cost

We notice in Figure 6(c) that for a large number of nodes, the computation cost of the SV is smaller than that of both the tree and gossip. In the latter, exactly one update operation is executed on a node at each round. For the situated scheme, the computation cost depends on the view size of each node, as in one round , an update operation is executed at each reception of a neighbour's value.

We observe a higher computation cost for the tree-based scheme because the worst case is registered at the root node where the computation cost is equal to the number of nodes. This phenomenon is due to the basic mechanism we have considered in which each node of the tree that receives a message from a child computes a partial aggregate and directly sends it to its parent. This result has to be carefully interpreted because it could let the reader understand that any form of tree-based approach would lead to such an explosion of the computation cost at the root, which does not reflect existing systems. However, in our case study, we have deliberately chosen to remove from the three evaluated protocols any threshold or synchronization mechanisms that would have made them more complex. We have opted for implementing and comparing only the most basic version of approaches to be able to identify the intrinsic limits they bring. As a consequence, our version of the tree protocol brings this high computation cost at the root. But, obviously, considering a tree-based aggregation scheme for an actual deployment would have to be coupled with a limitation mechanism, inversely increasing according to the distance from the root. To that aim, in threshold mechanisms embedded in nodes, avoiding communication of too much information on their uplink would be necessary to ensure the balance of the computing cost among the different autonomic managers. Finally, regarding the situated scheme, its low computing cost allows it to ensure the best reactivity.

6.2.4. Accuracy of estimated aggregates

We propose here to evaluate the accuracy of the estimated aggregates. To that aim, we used both the static and dynamic scenarios. In the first one, we fix the size of the network to 1000 nodes, and we measure the variance over the partial nodes' aggregates after each cycle of duration 200 ms.

We see in Figure 7(a) that the variance between the distributed aggregates decreases with an increase in the number of cycles. For the global schemes, the variance reaches 0 when all nodes converge to a unique global aggregate.[2] But for the situated scheme, the minimal variance is always greater than 0, showing that the situated scheme never converges towards the exact aggregated value. The numerical values of the variance once the aggregation process of situated scheme is finished are respectively 100 and 10 for SV1 and SV2. These values clearly show that SV1 provides an aggregated estimation that

[2] *Because of the log scale, the null value cannot be represented, and we choose to restrict the view in Figure 7(a) to the upper part of the curve to clearly see the difference between the variance values of situated and global schemes.*

DOI: 10.1002/nem

does not reflect the actual value at all. Thus, this scheme cannot act as a substitute of a global scheme providing a better convergence time. Its use has to be restricted either to management information that is effectively located and whose relevance is actually bound to a situated location or to management information that has a similar value on all nodes and that changes across the entire system in a similar way. By contrast, SV2 could act as a trade-off solution between situated and global schemes, as its variance is 10 times lower than that of SV1, showing that it could be a potential solution for a rapid estimation of global information.

By varying the information dynamics level within a static network of 1000 nodes, we obtained the results shown in Figure 8(a). We see that, with a deviation that reaches 2 when $\alpha = 5$, the tree is about five times more accurate than SV1, about three times more than SV2 and almost twice more than gossip. Thus, the fast convergence time of the situated scheme prevents it from suffering from the information dynamics. However, its partial computation still brings an important deviation from the actual aggregated value. As for global schemes, tree-based aggregation supports the information dynamics well owing to its fast convergence time. Nonetheless, as explained in [14], gossip is more sensitive to the information dynamics than the tree, exhibiting a higher deviation.

When we consider random fixed values on nodes and we vary the nodes' lifetime that is inversely proportional to the network churn level, we obtain the deviation measures of Figure 8(b). We notice that in the case of the tree-based scheme, the network dynamics dramatically affects the accuracy of estimated aggregates with a deviation that reaches 24. This sensibility is caused by the departures and arrivals of nodes, causing the continuous maintenance and reconstruction of the tree, whatever

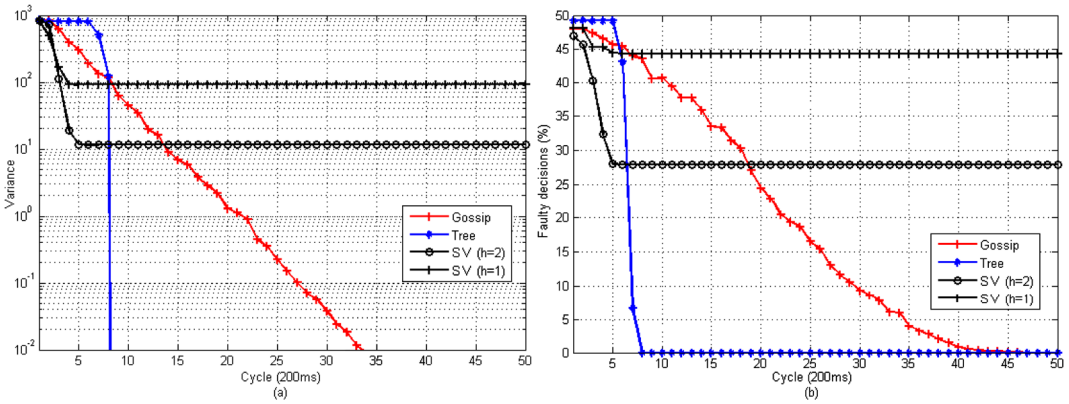


Figure 7. Performance results for the static scenario: (a) accuracy and (b) decision-making (semi-log scale)

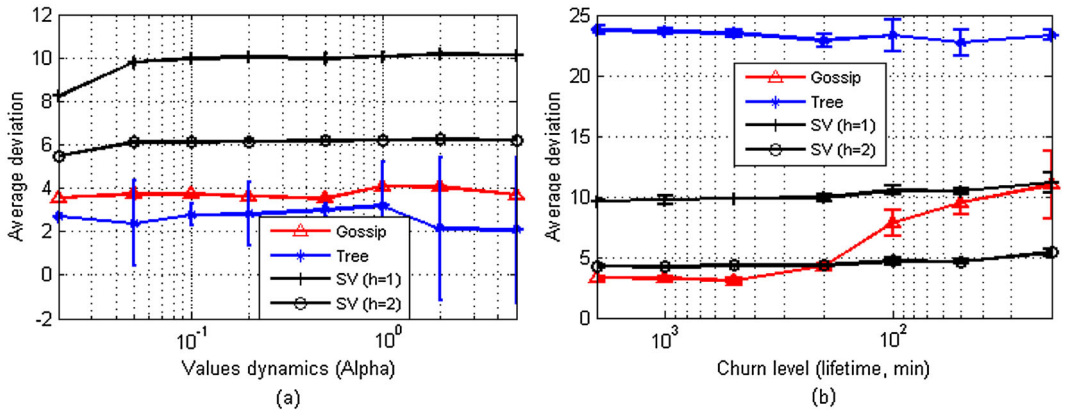


Figure 8. Accuracy of estimated aggregates for the dynamic scenario (semi-log scale)

DOI: 10.1002/nem

the churn level is. The situated scheme is more resilient to churn than gossip and tree because we register the same deviation for it in the case of static and dynamic environments. Thus, SV2 is about four times more accurate than the tree and about twice more than SV1. This is explained by the fact that, in the situated scheme, a departure or an arrival of a node does not necessarily affect other nodes if it is not in their view. Consequently, the parameter that controls the depth of this view in respect to the network size impacts on the churn robustness. Concerning the gossip scheme, the deviation is proportional to the churn level because it is a community process where nodes are dependent upon the correct behaviour of all other nodes. However, it is still the most accurate scheme under a low churn level where the average lifetime and inter-arrival are respectively higher than 200 min and 24 s. In the other interval, SV2 is the most accurate, acting as a good alternative to global schemes in highly dynamic environments.

6.2.5. *Efficiency of threshold-crossing detection*

As shown in Figure 7(b), for the static scenario, once converged, global schemes lead to exact decisions with a null number of faulty decisions. One can remark that reaching this value is four times longer for the gossip scheme than for the tree one. By contrast, the lack of accuracy of the situated scheme directly impacts the decision-making part of AMs. The worst case appears for SV1 with 45% of faulty decisions, clearly showing that a one-hop situated scheme is unable to infer a global network state. If SV2 could be considered as a trade-off between global schemes and SV1 regarding its accuracy and convergence time, the ratio of faulty decisions, close to 28%, shows that it cannot finally replace a global scheme for a managed system-wide inference.

Regarding the dynamic scenario, we observe in Figure 9(a) that the number of faulty decisions increases according to the information dynamics level. The information dynamics affects the aggregation process and then the decision-making one. One can note that for all the aggregation schemes, the number of faulty decisions ranges in [5; 12]% when $\alpha = 0.02$, but when $\alpha = 1$ or 2, the number of faulty decisions reaches the interval [45; 50]% where the decision-making process is unable to take the right decisions. Globally, this error percentage is larger in the SV, while gossip offers the best quality of the decision-making. Under churn, Figure 9(b) shows that, in the case of gossip and SV2, the number of faulty decisions is comparable and lower than the one registered when using a tree or SV1. Moreover, it is proportional to the churn level. We also observe that churn hugely affects the quality of the decision-making process on the tree where the average number of faulty decisions is about 45%. Thus, it is more relevant to use gossip or SV2 on a dynamic network.

6.3. *Synthesis*

Given the set of results we obtained through the simulation campaign presented earlier, we are able to identify the context in a qualitative and quantitative manner, regarding both the behaviour of the

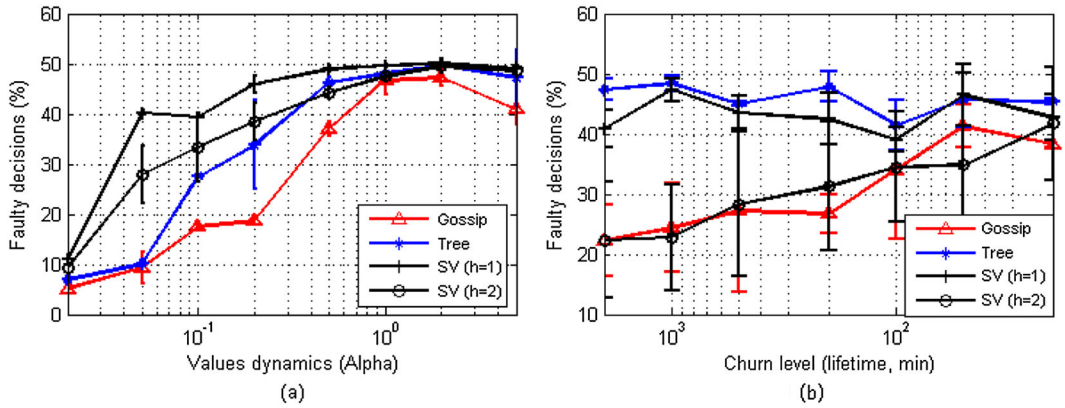


Figure 9. Efficiency of the decision-making for the dynamic scenario

DOI: 10.1002/nem

aggregated information and the operational context, in which an aggregation scheme outperforms the others. More specifically, through the help of numerical factors, we first highlight the quantitative differences between aggregation schemes. Now, we abstract this analysis by introducing a qualitative synthesis, based on abstract scores, that highlight the area in which a scheme outperforms the others.

The situated scheme is intended to design very reactive autonomic management frameworks able to quickly obtain and process an information situated in a direct neighbourhood even in a very dynamic environment. For example, regarding the convergence time, that of SV1 is about five times lower than that of tree and 33 times lower than that of gossip, enabling it to deal with dynamic information. This reactivity is also applied to the computation cost that is lowest. Moreover, because only a subset of nodes belonging to a managed system is involved in the aggregation process, the churn has no significative impact on it. By contrast and for the same reason, the situated scheme is not accurate for system-wide information, especially for SV1. Thus, the choice of the aggregated information must be carefully studied to ensure that it only stands for an actual situated subset of the managed domain.

The case for SV2 is a trade-off between situated and global schemes. Its accuracy is better than that of SV1, but its very high communication cost could be prohibitive in constraint environments: the communication cost of SV2 can be almost three times higher than the one obtained in the case of gossip and about 42 times that of tree.

Global aggregation schemes offer the best accuracy because they consider all the available information in the aggregation process, which directly impacts the quality of the decision-making process, in case of a threshold detection. Especially, the tree-based scheme presents the fastest convergence time, enabling it to aggregate dynamic information with an excellent accuracy (we measured a deviation that reaches 2 when $\alpha = 5$). It is about five times more accurate than SV1, about three times more than SV2 and almost twice more than gossip. Nonetheless, this scheme is dramatically affected by churn because even in the presence of a low or moderate churn, the average number of faulty decisions reaches 45%. Regarding the communication cost, the tree-based scheme provides the best option.

Finally, the gossip scheme presents the longest convergence time while being resilient to churn. If such a convergence time makes it unable to deal with highly dynamic information, in the presence of an acceptable churn level, gossip performs better than the other schemes in terms of aggregation accuracy and efficiency of the decision-making process, followed by the SV and then the tree. More precisely, when the average lifetime and inter-arrival are respectively higher than 200 min and 24 s, gossip is about four times more accurate than the tree and about twice more than SV1, with a comparable accuracy with SV2. However, when the average lifetime and inter-arrival are respectively lower than 200 min and 24 s, gossip has a comparable accuracy with that of SV1, and SV2 becomes the most accurate scheme.

Table 5 summarizes the obtained quantitative simulation results. We attribute a score for each aggregation protocol according to its performance in the current execution context. This score can have one of five different values classed from the worst one to the best one : $--$, $-$, $=$, $+$ and $++$. It can serve as a basis towards the design of rules guiding the choice of an aggregation scheme in an autonomic management system.

When comparing these results with those found in the literature and presented in Section 2, we corroborate the fact that tree-based approaches are very cheap in terms of message exchanges but sensitive to churn and link failures. Moreover, they present the best accuracy to be effective in network management. As already highlighted in the literature, gossip-based approaches are slower but cheaper and more churn resilient than tree-based approaches. Finally, we corroborate the fact that SVs are very fast, fault tolerant but unprecise. However, we also found that SV has a higher communication cost than gossip-based approaches.

7. CONCLUSION AND FUTURE WORK

In this paper, we have presented an evaluation study we have performed to propose generic criteria that could help management system designers in choosing the best approach for a given context. We have proposed a refined taxonomy of aggregation schemes that augment the standard network structure and propagation technique criteria to catch features that finely differentiate them. We instantiated

DOI: 10.1002/nem

Table 5. Abstracted simulation results

	Convergence time	Communication cost	Computation cost	Aggregation accuracy	Churn support	Information dynamics support
Tree	=	++	−−	++	−−	+
Gossip	−−	=	=	+	=	−
SV1	++	−	++	−−	++	++
SV2	+	−−	+	−	+	+

SV, situated view.

this taxonomy over existing proposals and presented for each approach a brief overview of its operations. From that point, we designated three schemes that are the most representative of each category. We implemented them in a simulation framework and evaluated them according to their performance and cost. Through this evaluation, we have highlighted operational contexts and type of information for which a scheme outperforms the others: 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 SV depends on the view size of nodes. Such a work could serve as a basis for the enlightened choice of an approach when designing new autonomic management systems.

Regarding the research perspectives, because network operational states are dynamic, we explore the possibility to let the management system, at each moment, decide by itself the best aggregation scheme to use. Such an approach would free management system designers from the choice of a static approach that would instead be dynamically determined at the production stage. Designing such a system is challenging because it requires the network to autonomously monitor itself and build a meta-management plane in which AMs could extract contextual information to set the best strategy to use. In this context, ensuring the stability of the management plane and dealing with transient phases when swapping from one scheme to another are strong challenges that we are currently addressing. Also, in an effort to enhance the given evaluation, we are looking for ways to consolidate this evaluation work. A short-term direction would lie in the consideration of additional impacting parameters (e.g. network topology model, changing frequency of local values, various neighbourhood degrees or the depth of the situated view). A long-term one would consist in the exhaustive study of the impact of monitoring schemes on the decision -making process in autonomic management. In this paper, we only introduced it through the evaluation of faulty decisions in a threshold-crossing detection context. But such a work would require the consideration of (i) more complex management data models that would for example rely on a nonlinear dynamics, (ii) additional impact metrics (e.g. distance from the threshold or the global reaction delay) and (iii) the consideration of other decision rules (e.g. use of hysteresis cycles for lower and upper thresholds). Finally, the approach we followed in our work is of an abstract type regarding both network and management information models. Thus, a straight next step would rely in the instantiation of our study in concrete management scenario, in other words for specific network models (e.g. P2P live streaming systems and social networks) and for data coming from their related management information models. Through that way, it would be possible to go one step beyond the guidelines we presented in the paper by proposing the best aggregation strategy for any couple of concrete managed networks and type of aggregated information. Software-defined networks (SDN) [84] with their ability to dynamically control network data planes could now offer a framework in which the results of this study could be implemented because they provide the expected capacities to support autonomic functions at the network level. SDN controllers tend to be distributed and even decentralized, and the design of autonomic monitoring solutions, which actually fits with the dynamic operational context they are deployed in, stands for a concrete case of applicability.

REFERENCES

1. Horn P. Autonomic Computing: IBM's Perspective on the State of Information Technology: IBM, 2001. Avialable: http://www.ibm.com/research/autonomic.
2. Kephart JO, Chess DM. The vision of autonomic computing. *Computer* 2003; **36**(1): 41–50.

DOI: 10.1002/nem

3. Müller-Schloer C, Hartmut Schmeck TU (eds.) *Organic Computing—A Paradigm Shift for Complex Systems.* Springer Basel: Germany, 2011.
4. Bauher B, Kasinger H. AOSE and organic computing—how can they benefit from each other? *Lecture Notes in Computer Science* 2005; **3770**: 109–118.
5. Schmeck H. Organic computing—vision and challenge for system design. In *4th International Conference on Parallel Computing in Electrical Engineering*, 2004; 3–3.
6. Agoulmine N. *Autonomic Network Management Principles: From Concepts to Applications.* Press Academic (ed.), Elsevier: Oxford, 2010.
7. Dobson S, Denazis S, Fernández A, Gaïti D, Gelenbe E, Massacci F, Nixon P, Saffre F, Schmidt N, Zambonelli F. A survey of autonomic communications. *ACM Transactions on Autonomous and Adaptive Systems* 2006; **1**(2): 223–259.
8. Bullot T, Khatoun R, Hugues L, Gaïti D, Merghem-Boulahia L. A situatedness-based knowledge plane for autonomic networking. *International Journal of Network Management* 2008; **18**(2): 171–193.
9. Makhloufi R, Bonnet G, Doyen G, Gaïti D. Decentralized aggregation protocols in peer-to-peer networks: a survey. In *4th IEEE International Workshop on Modelling Autonomic Communications Environments*, 2009; 111–116.
10. Makhloufi R, Doyen G, Bonnet G, Gaïti D. Situated vs. global aggregation schemes for autonomous management systems. In *4th IFIP/IEEE International Workshop on Distributed Autonomous Network Management Systems*, 2011; 1131–1135.
11. Makhloufi R, Doyen G, Bonnet G, Gaïti D. Impact of dynamics on situated and global aggregation schemes. In *5th IFIP International Conference on Autonomous Infrastructure, Management, and Security*, 2011; 148–159.
12. Bawa M, Garcia-Molina H, Gionis A, Motwani R. Estimating aggregates on a peer-to-peer network. *Technical Report*, Stanford InfoLab, 2003.
13. Wuhib F, Dam M, Stadler R, Clem A. Robust monitoring of network-wide aggregates through gossiping. *IEEE Transactions on Network and Service Management* 2009a; **6**(2): 95–109.
14. Birman K. The promise, and limitations, of gossip protocols. *ACM SIGOPS Operating Systems Review* 2007; **41**(5): 8–13.
15. Jesus P, Baquero C, Almeida PS. A survey of distributed data aggregation algorithms. *Technical Report*, University of Minho, 2011.
16. Stadler R. Protocols for distributed embedded management. In *Network-embedded Management and Applications,* Springer: New York, 2013, pp. 263–290.
17. Krishnamachari B, Estrin D, Wicker S. The impact of data aggregation in wireless sensor networks. In *22nd International Conference on Distributed Computing Systems Workshops*, 2002; 575–578.
18. Rajagopalan R, Varshneys PK. Data-aggregation techniques in sensor networks: a survey. *IEEE Communications Surveys and Tutorials* 2006; **8**(1–4): 48–63.
19. Sang Y, Shen H, Inoguchi Y, Tan Y, Xiong N. Secure data aggregation in wireless sensor networks: a survey. In *7th International Conference on Parallel and Distributed Computing, Applications and Technologies*, 2006; 315–320.
20. Fasolo E, Rossi M, Widmer J, Zorzi M. In-network aggregation techniques for wireless sensor networks: a survey. *IEEE Wireless Communications* 2007; **14**(2): 70–87.
21. Nakamura EF, Loureiro AAF, Frery AC. Information fusion for wireless sensor networks: methods, models, and classifications. *ACM Computer Survey* 2007; **39**(3): 1–55.
22. Alzaid H, Foo E, Nieto JG. Secure data aggregation in wireless sensor network: a survey. In *6th Australasian Conference on Information Security*, 2008; 93–105.
23. Ozdemir S, Xiao Y. Secure data aggregation in wireless sensor networks: a comprehensive overview. *Computer Network* 2009; **53**(12): 2022–2037.
24. Pandey V, Kaur A, Chand N. A review on data aggregation techniques in wireless sensor network. *Journal of Electric and Electrical Engineering* 2006; **1**(2): 1–8.
25. Goryczka S, Xiong L, Sunderam V. Secure multiparty aggregation with differential privacy: a comparative study. In *EDBT/ICDT Workshops*, 2013; 155–163.
26. Jose J, Princy M, Jose J. Integrity protecting and privacy preserving data aggregation protocols in wireless sensor networks: a survey. *International Journal on Computer Network and Information Security* 2013; **7**: 66–74.
27. Spandan G, Patel A, Manjunath C-R, Nagaraj G. Data aggregation protocols in wireless sensor networks. *International Journal of Computational Engineering Research* 2013; **3**(5): 18–24.
28. Kempe D, Dobra A, Gehrke J. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science*, 2003; 482–487.
29. Kesselman C, Foster I. *The Grid: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers: Francisco, CA, USA, 1998.
30. Subramanian R, Goodman BD. *Peer to Peer Computing: The Evolution of a Disruptive Technology.* Idea Group Publishing: Hershey, PA, USA, 2005.
31. Akyildiz I, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. *Computer Networks* 2002; **38**(4): 393–422.
32. Grabisch M, Marichal J-L, Mesiar R, Pap E. *Aggregation Functions (Encyclopedia of Mathematics and Its Applications).* Cambridge University Press, 2009.
33. Sarkar R, Zhu X, Gao J. Hierarchical spatial gossip for multi-resolution representations in sensor networks. In *6th International Conference on Information Processing in Sensor Networks*, 2007; 420–429.

DOI: 10.1002/nem

34. Flajolet P, Martin GN. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences* 1985; **31**(2): 182–209.
35. Jurca D, Stadler R. Computing histograms of local variables for real-time monitoring using aggregation trees. In *11th IFIP/IEEE International Conference on Symposium on Integrated Network Management*, 2009; 367–374.
36. Rabbat MG. On spatial gossip algorithms for average consensus. In *14th IEEE/SP Workshop on Statistical Signal Processing*, 2007; 705–709.
37. Xiao L, Boyd S, Lall S. A scheme for robust distributed sensor fusion based on average consensus. In *3rd International Symposium on Information Processing in Sensor Networks*, 2005; 63–70.
38. Kempe D, McSherry F. A decentralized algorithm for spectral analysis. In *36th Annual ACM Symposium on Theory of Computing*, 2004; 561–568.
39. Rabbat M, Haupt J, Singh A, Nowak R. Decentralized compression and predistribution via randomized gossiping. In *4th International Symposium on Information Processing in Sensor Networks*, 2006; 51–59.
40. Renesse RV, Birman KP, Vogels W. Astrolabe: a robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems* 2003; **21**(2): 164–206.
41. Montresor A, Jelasity M, Babaoglu O. Robust aggregation protocols for large-scale overlay networks. In *International Conference on Dependable Systems and Networks*, 2004; 1–4.
42. Madden S, Franklin MJ, Hellerstein JM, Hong W. TAG: a tiny aggregation service for ad-hoc sensor networks. *ACM SIGOPS Operating Systems Review* 2002; **36**(SI): 131–146.
43. Gray J, Bosworth A, Layman A, Pirahesh H. Data cube: a relational aggregation operator generalizing group-by, cross-tab, and sub-total. In *12th International Conference on Data Engineering*, 1996; 152–159.
44. Barbará D, DuMouchel W, Faloutsos C, Haas PJ, Hellerstein JM, Ioannidis Y, Jagadish HV, Johnson T, Ng R, Poosala V, Ross KA, Sevcik KC. The New Jersey data reduction report. *IEEE Data Engineering Bulletin* 1997; **20**(4): 3–45.
45. Dam M, Stadler R. A generic protocol for network state aggregation. In *Radiovetenskap och Kommunikation,* Stockholm, 2005.
46. Li J, Yoh Lim D. A robust aggregation tree on distributed hash tables. In *MIT Student Oxygen,* MIT Student Alliance: Cambridge, MA, USA, 2004.
47. Dietzfelbinger M. Gossiping and broadcasting versus computing functions in networks. *Discrete Applied Mathematics* 2004; **137**(2): 127–153.
48. Jelasity M, Montresor A, Babaoglu O. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems* 2005; **23**(3): 219–252.
49. Meshkovaa E, Riihijärvia J, Petrovaa M, Mähönen P. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Computer Networks* 2008; **52**(11): 2097–2128.
50. Sacha J, Napper J, Stratan C, Pierre G. Adam2: reliable distribution estimation in decentralised environments. In *30th International Conference on Distributed Computing Systems*, 2010; 697–707.
51. Prieto AG, Stadler R. A-GAP: an adaptive protocol for continuous network monitoring with accuracy objectives. *IEEE Transactions on Network and Service Management* 2007; **4**(1): 2–12.
52. Binzenhöfer A, Kunzmann G, Henjes R. A scalable algorithm to monitor Chord-based P2P systems at runtime. In *IEEE International Parallel and Distributed Processing Symposium*, 2006; 239–249.
53. Artigas MS, López PG, Gómez-Skarmeta AF. DECA: a hierarchical framework for decentralized aggregation in DHTs. In *International Workshop on Distributed Systems: Operations and Management*, 2006; 246–257.
54. Villas L, Boukerche A, Ramos H, de Oliveira H, de Araujo R, Loureiro A. DRINA: a lightweight and reliable routing approach for in-network aggregation in wireless sensor networks. *IEEE Transactions on Computers* 2013; **62**(4): 676–689.
55. Baquero C, Almeida PS, Menezes R, Jesus P. Extrema propagation: fast distributed estimation of sums and network sizes. *IEEE Transactions on Parallel and Distributed Systems* 2012; **23**(4): 668–675.
56. Al-Karaki JN, Ul-Mustafa R, Kamal AE. Data aggregation and routing in wireless sensor networks: optimal and heuristic algorithms. *Computer Network* 2009; **53**(7): 945–960.
57. Haridasan M, van Renesse R. Gossip-based distribution estimation in peer-to-peer networks. In *7th International Workshop on Peer-to-Peer Systems*, 2008; 13–18.
58. Li Z, Peng Y, Qiao D, Zhang W. LBA: lifetime balanced data aggregation in low duty cycle sensor networks. In *31st Annual IEEE International Conference on Computer Communications*, 2012; 1844–1852.
59. Geibig J, Bradler D. Self-organized aggregation in irregular wireless networks. In *Wireless Days*, 2010; 1–7.
60. Wuhib F, Stadler R. M-GAP: a new pattern for cfengine and other distributed software. *Technical Report*, Royal Institute of Technology (KTH), 2008.
61. Boyd S, Ghosh A, Prabhakar B, Shah D. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking* 2006; **14**(SI): 2508–2530.
62. Yalagandula P, Dahlin M. A scalable distributed information management system. *ACM SIGCOMM Computer Communication Review* 2004; **34**(4): 379–390.
63. Sen J. A robust and secure aggregation protocol for wireless sensor networks. In *6th International Symposium on Electronic Design, Test and Application*, 2011; 222–227.
64. Wuhib F, Dam M, Stadler R, Clem A. Robust monitoring of network-wide aggregates through gossiping. *IEEE Transactions on Network and Service Management* 2009b; **6**(2): 95–109.
65. Wuhib F, Dam M, Stadler R. Decentralized detection of global threshold crossings using aggregation trees. *Computer Networks* 2008; **52**(9): 1745–1761.

DOI: 10.1002/nem

66. Renesse R, Bozdog A. Willow: DHT, aggregation, and publish/subscribe in one protocol. In *Peer-to-Peer Systems III,* Springer LNCS: Berlin, Heidelberg, 2005, pp. 173–183.
67. Lim K, Stadler R. A navigation pattern for scalable internet management. In *7th International Symposium on Integrated Network Management*, 2001; 405–420.
68. Sancho J, Robles A, Duato J. A flexible routing scheme for networks of workstations. In *High Performance Computing*, Valero M, Joe K, Kitsuregawa M, Tanaka H (eds.), Lecture Notes in Computer Science, vol. 1940 Springer: Berlin Heidelberg, 2000, pp. 260–267.
69. Kempe D, Kleinberg J, Demers A. Spatial gossip and resource location protocols. In *33rd Annual ACM Symposium on Theory of Computing*, 2001; 163–172.
70. Lin S, Taïani F, Blair GS. Gossipkit: a framework of gossip protocol family. In *5th Middleware for Network Eccentric and Mobile Applications Workshop*, 2007; 26–30.
71. Dimakis A, Sarwate A, Wainwright M. Geographic gossip: efficient aggregation for sensor networks. In *5th International Conference on Information Processing in Sensor Networks*, 2006; 69–76.
72. Aggarwal CC, Yu PS. *Data Streams: Models and Algorithms.* A survey of synopsis construction in data streams, Springer-Verlag: New York, 2006.
73. Gouvas P, Zafeiropoulos A, Liakopoulos A. Gossiping for autonomic estimation of network-based parameters in dynamic environments. In *International Conference on the Move to Meaningful Internet Systems*, 2010; 358–366.
74. Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H. Chord: a scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, 2001; 149–160.
75. Makhloufi R, Bonnet G, Doyen G, Gaïti D. Towards a P2P-based deployment of network management information. In *4th IFIP International Conference on Autonomous Infrastructure, Management, and Security*, 2010; 26–37.
76. Leitao J, Pereira J, Rodrigues L. Large-scale peer-to-peer autonomic monitoring. In *3rd IEEE Workshop on Distributed Autonomous Network Management Systems*, 2008; 1–5.
77. Wuhib F, Dam M, Stadler R, Clemm A. Robust monitoring of network-wide aggregates through gossiping. In *10th IFIP/IEEE International Symposium on Integrated Management*, 2007; 226–235.
78. Leonard D, Rai V, Loguinov D. On lifetime-based node failure and stochastic resilience of decentralized peer-to-peer networks. *Sigmetrics* 2005; **33**: 26–37.
79. Li J, Stribling J, Morris R, Kaashoek MF, Gil TM. A performance vs. cost framework for evaluating DHT design tradeoffs under churn. In *IEEE International Conference on Computer Communications*, 2005; 225–236.
80. Rhea S, Geels D, Roscoe T, Kubiatowicz J. Handling churn in a DHT. In *Annual Conference on USENIX Annual Technical Conference*, 2004; 1127–140.
81. Rowstron AIT, Druschel P. Pastry: scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms*, 2001; 329–350.
82. Castro M, Druschel P, Kermarrec A-M, Rowstron A. Scribe: a large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications* 2002; **20**(8): 1489–1499.
83. Barabási A, Albert R. Emergence of scaling in random networks. *Science* 1999; **286**(5439): 509–512.
84. McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, Shenker S, Turner J. OpenFlow: enabling innovation in campus networks. *SIGCOMM Computer Communication Review* 2008; **38**(2): 69–74.

AUTHORS' BIOGRAPHIES

Rafik Makhloufi is currently a CTO manager at Optiways in Paris, France. He received his PhD in Computer Science on 2012 from Troyes University of Technology. He obtained his Master's and engineer degrees in Computer Science, respectively, on 2008 and 2007 from the University of Paris-Sud 11 (France) and University of Bejaia (Algeria). After his PhD, he worked in different research institutions like Troyes University of Technology, University of Reims Champagne-Ardenne and Ecole Nationale des Ponts et Chaussées. His recent research interests include transport, network management, ad hoc and sensor networks, security, performance evaluation and simulation.

Guillaume Doyen has been an associate professor in Troyes University of Technology since 2006. He is affiliated with both the CNRS Charles Delaunay Institute (UMR CNRS 6281) and the INRIA Grand Est as an associate researcher. His current research interest focuses on the design of autonomous management and control solutions applied to the performance and security of content distribution and cloud computing. He is actively involved in the network and service management community (TPC co-chair of the IFIP AIMS conference in 2013 and 2014, TPC member of the IFIP/IEEE NOMS and IM conferences since 2011 and also reviewer for IJNM, JNSM and TNSM journals).

DOI: 10.1002/nem

Grégory Bonnet received his PhD in Computer Science from the University of Toulouse in 2008. He has been a postdoctoral Fellow from 2009 to 2010 at the University of Technology of Troyes, applying Artificial Intelligence techniques to Autonomic Network Management. He has been an assistant professor at the University of Caen Lower-Normandy since 2010. His research areas include multiagent systems, coalition formation and trust manipulation in reputation systems. He serves on the organization committee of the 20th French National Conference on Multi-Agent System. Since 2014, he has been the coordinator of the Ethics and Autonomous Agents project, which is partially funded by the French National Research Agency.

Dominique Gaïti received her PhD and 'Habilitation à diriger des recherches' degrees in Computer Science from the University of Paris VI and Paris IX on 1991 and 1995, respectively. She is currently a professor at the University of Technology in Troyes (France), a member of the Institute Charles Delaunay (ICD UMR 6281). She is the leader of the team 'autonomic networking' in this institute. She was previously a research scientist at the University of Columbia (New York, USA), 1992-1994 and a researcher at the University of Paris 6, member of the LIP6 laboratory (Paris, France), 1996-1997. She was the chairperson of the IFIP WG 6.7 on 'smart networks' during 6 years. Her research interests include the smart networks, the intelligence in networks, and the control and management (through intelligent agents) in all types of networks. She is the an author of one book and has edited several proceedings.

DOI: 10.1002/nem