NOA-AID: Network Overlays for Adaptive Information Aggregation, Indexing and Discovery at the Edge

Patrizio Dazzi[1](✉) and Matteo Mordacchini[2]

[1] ISTI–CNR, Pisa, Italy
patrizio.dazzi@isti.cnr.it
[2] IIT–CNR, Pisa, Italy
matteo.mordacchini@iit.cnr.it

Abstract. This paper presents NOA-AID a network architecture for targeting highly distributed systems, composed of a large set of distributed stream processing devices, aimed at adaptive information indexing, aggregation and discovery in streams of data. The architecture is organized on two layers. The upper layer is aimed at supporting the information discovery process by providing a distributed index structure. The lower layer is mainly devoted to resource aggregation based on epidemic protocols targeting highly distributed and dynamic scenarios, well suited to stream-oriented scenarios. We present a theoretical study on the costs of information management operations, also giving an empirical validation of such findings. Finally, we presented an experimental evaluation of the ability of our solution to be effective and efficient in retrieving meaningful information in streams on a highly-dynamic and distributed scenario.

Keywords: IoT · Stream · Adaptivity · Network overlay
Information aggregation in streams · Distributed indexing

1 Introduction

In recent times we are witnessing the emergence of pervasive computational environments in which a huge amount of distributed and heterogeneous devices produce, transmit and/or observe continuous streams of data. Such streams of data needs to be processed to detect faults, issue alerts, and trigger management operations. To achieve an efficient analysis of such data, it is gaining momentum the exploitation of high-performance solutions tailored on recent commodity parallel hardware and accelerators typically available on modern IoT and Edge devices. Even more recently, an increasing interest is coagulating around the methodologies enabling a fruitful cooperation of such devices, which are no longer limited to be independent stream processing entities but pieces of a complex and distributed system. Efficient and effective communication supports for information gathering, exchange, indexing and querying are of paramount importance in this context. As matter of facts, every information discovery process is strongly correlated to its query formulation and resolution mechanism. The query formulation process has to support an effective way to express needs, whereas the query resolution mechanism must be able to leverage the query expressiveness to efficiently find the information requested and to limit the overhead introduced by the process itself. A common technique for finding data and information, in a highly distributed and dynamic scenario is based on range queries over a set of different attributes [1–5]. However, the heterogeneous nature of distributed devices and the high dynamicity characterizing the information belonging to streams, often makes the task of query formulation very complex. For instance when an information is defined as the combination of many different attributes, it could not be easy to identify the most relevant and discriminating features. An interesting alternative consists in defining a *simulacrum*, representing the archetype of the information sought. This provides to the search system a mean to identify the desired set of information into data streams whilst relieving the requester from specifying complex queries. To this end, the discovery system needs to be organised accordingly. First, there is a need for a search system supporting approximated searches on data streams, enabling the system to deliver the best match against the provided simulacrum. Second, "information providers", which in our case are IoT or Edge devices devoted to stream processing, need an efficient discovery infrastructure, i.e., characterised by a reduced cost of maintenance, while ensuring that information can be easily found by requesters.

To date, several solutions, have been focusing on such approach. They try to let nodes to self-organise to disseminate the information toward groups of interested nodes [6] and/or they let each node to be in direct contact with the ones having similar data [7–12]. However, the local knowledge maintained by each device usually does not allow a proper identification of the features which characterise an entire community of nodes sharing a common set of information. Many existing approaches rely only on the information that each device owns, without providing any explicit identification of groups of nodes that can be considered as a community. This work presents a distributed architecture organised on two layers providing: (i) a flexible query-by-example (the aforementioned simulacrum) discovery mechanism and (ii) a solution for stream processing devices easing the information advertisement process. The focus is on scenarios in which IoT and Edge devices composing the discovery system consist in entities called *Advertising Nodes* (AN). Each AN has an associated succinct description of the information observed by such device: its *profile*. The proposed solution couples the flexibility of unstructured overlays with the power of structured networks. The former offer the advantage of a low maintenance cost, whereas the latter offer more guarantees on finding the requested resources but at the cost of a more expansive maintenance. The rest of this paper is organised as follows. Section 2 presents a review of the relevant literature. Section 3 presents the overall architecture of NOA-AID. Sections 4 and 5 describe the unstructured- and structured-layer, respectively. Section 6 presents the conducted evaluation. Finally, conclusions are given in Sect. 7.

2 Background and Related Work

The challenge of searching for information in highly distributed environment is very current and relevant. In spite of this, many work has been proposed so far. In this section we report some of the most relevant approaches facing this challenge. Multi-Attribute Addressable Network (MAAN) [1] consists in a structured system able to support multi-attribute range queries. In MAAN, items are identified by a set of attribute-value pairs, and each attribute is mapped on a bucket through a locality preserving function. The node target of such function stores the full item description so that each item is stored as many times as the number of its attributes. The resolution of a multi-attribute range query consists in executing a single one-dimensional query on the dominant (i.e. most selective) attribute, while the other attributes are checked using the replicated data. Although MAAN provides a smart routing technique and it has the ability to perform queries on subsets of the whole attributes domain, it requires large amount of memory to store resource indices, and a high computational cost to maintain them up-to-date. This class of solutions requires users to be aware of all the indexed attributes and their respective domains. Making queries exploiting only a small subset of them without specifying the other ones, or not defining the attributes range properly, it may happen that too many results are returned leading the user to iteratively refine her/his queries. More flexible queries can be expressed in DHT-based systems. MCAN [13], exploits the CAN architecture, where, in each dimension, coordinates are given by the distance from a given pivot. Although such solutions allow users to exploit the query-by-example paradigm, these proposals are other examples of ad-hoc solutions, though to be used for searching multimedia objects, and thus are unsuitable for more generic kinds of resources. Pirrò *et al.* [14] show an approach for a semantic-based service discovery in P2P networks. It couples a DHT layer with a SON (Semantic Overlay Network) overlay. Differently from our solution, here a DHT-based network allows peers to publish semantically annotated services. Then a SON is incrementally build by using the interactions between peers within the structured level during the service publication and searching processes. In our solution only the community representatives are registered in the DHT level. To have only a subset of the devices composing the network in the DHT layer leads to reduce the number of messages routed through the DHT to solve a query. GosSkip [15] is a self-organizing and fully distributed gossip overlay that provides a support to data storage and retrieval in highly decentralized environments. It is built using a epidemic protocol that organizes peers to form an ordered double-linked list. In the overlay network each peer is connected in a skip list where connection are similarity based. To this end, each node is associated with a single item of data and it has a name that describes the semantics of the associated object. These names follow a total and deterministic order. As a consequence, the position of an element is fully determined by its name. For information dissemination, its gossip protocol maintains $O(log(N))$ peer states, and has a message routing cost of $O(log(N))$. The association of links to the published object can lead to a very large number of connections. This is especially true in networks where the number of objects shared by each node is large. The main drawback of the above solutions is the lack of a broader, more recognized measure of similarity. Each peer only relies on its local view. Thus, it is not able to determine whether a peer not included in its similarity-based neighbourhood could be regarded as similar with respect to the overall network organization. More effective information dissemination cannot be implemented because peers are not able to determine whether or not there exist between them more latent forms of similarity, even when they do not consider each other as immediate neighbours. Another type of unstructured networks organization is given by Semantic Overlay Networks (SON). Crespo and Garcia-Molina [16] organize peers in clusters of semantically correlated nodes, on the basis of the semantic content of the document they share. Each cluster represent a semantic concept, i.e., peers belong to groups that go beyond their simple neighbourhood. The assignment of peers (and queries) to a given cluster is made using a hierarchical classifier organised as a tree, where each node is a concept. Nodes encountered descending such a tree represent semantic refinements of the concept of their father nodes. A SON is created for each node of tree of concepts. The main disadvantages of this class of solutions is the rigid predefined structure of the SON-based overlay network. Crespo and Garcia-Molina [16] assume that the concept of tree is pre-defined and peers must use the same classifier in order to join a group. We seek to create more dynamic, spontaneous communities, dynamically made by the interactions between nodes and without relying on a priori knowledge on how to classify the shared content.

3 Overall Architecture

The overall architecture of our proposed solution organized on two layers (structured and unstructured networks) and four different kinds of entities (advertising node, community representative, node belonging to the structured layer and requesters) realising the NOA-AID ecosystem. The unstructured layer is based on an highly scalable epidemic protocol, whereas the structured network is based on a properly defined Distributed Hash Table (DHT). The structured layer indexes profiles of Community Representatives (CR). Each CR is elected by a community of Advertising Nodes (AN). Each AN has an associated *profile*, i.e., a set of information continuously extracted during the stream processing phase. The unstructured layer is devoted to build communities by means of a similarity function applied on the profiles describing the data passed through of streaming processing devices. Each community elects its own representative, which is in charge of registering itself on the structured layer, that is the layer to query in order to search for the information sought. The query resolution process is organised on two stages. Firstly, it is queried the structured layer providing it a *simulacrum* of the information searched. This layer returns the CRs that are the closer to the simulacrum. Then, the selected CRs, acting as entry points, percolate the queries inside their own communities to search for ANs that actually satisfy the needs expressed by the query. Profiles are used to compare the information associated to different devices. To build profiles of streaming processing devices, many different functions and profile organizations can be exploited. This both depends on the ultimate aim of the system and on the type of information to manage and index. In this work we assume that the streams are made of textual data, thus the profiles represent collections of words. To measure the similarity between two different profiles, we adopted a slightly modified version of the Jaccard similarity coefficient [17], described in Sect. 5. It has proven to be an effective measure in distributed environments [18]. It is computed as the size of the intersection of two sets divided by the size of their union. However, traditional DHTs, providing mechanisms for exact matches, are not efficient for searching resources in highly heterogeneous and dynamic scenarios. To overcome this limitation, we instrumented our structured network to perform approximate matches between a user query and a community profile. To this end we leveraged a Locality Sensitive Hash (LSH) method. An appropriate representation of profiles is important to tailor an information discovery mechanism to a specific aim or application. However, such investigation is beyond the scope of this paper. In our study we limit our investigation to two profile representation:

- *Weighted Attribute Vector*: a collection of words, weighted according to their relevance with respect to a profile.
- *Attribute Adjacency Matrix* (hereafter Adjacency Matrix): a profile is represented with a weighted word of vector enriched with values estimating the correlation between attributes.

Among the two, the Weighted Attribute Vector is the simplest. It contains all the attributes describing the stream observed by a node along with their relevance weight values. Since it is a composition of all the attributes, the relevance weight value should be computed by taking into consideration all the single attribute values extracted from the stream. The exploitation of the Adjacency Matrix as profiles permits to represent a relational graph between attributes by using the co-occurrences of them in the set of information represented by a device. Each row of the resulting matrix is associated to an attribute $Attr_i$. Each entry j of such a row contains the co-occurrences proportion of $Attr_i$ with an attribute $Attr_j$. The i-th entry of the $Attr_i$'s row simply gives the relevance value associated to $Attr_i$. Entries are zero-valued when there is no relation between the referred pair of attributes.

4 The Unstructured Layer

The lower layer is aimed at the detection and the creation of self-emerging communities made up of Advertising Nodes. This layer is based on the GROUP protocol. GROUP is a protocol we conceived, designed and implemented for building communities in a completely decentralised way. An in-depth presentation of GROUP is beyond the goals of this paper. We refer interested readers to the original paper in which it has been presented and analyzed [19–21]. Here we briefly present its behaviour and approach. Group carries out communities of similar Advertising Nodes by achieving a logic partition $\mathscr{P}_I = \{P_1, \ldots, P_s\}$

of nodes belonging to a network, such that every P_i includes a subset of nodes characterised by similar profiles. Each distinct partition P_i represents a different community. To identify the communities GROUP exploits a distributed voting algorithm on the overlays built by other epidemic protocols. This process is driven by the consensus that a certain AN gathers among the other ANs. Each AN votes for the ANs it considers closer to itself, i.e. the ones with a profile similar to its own. Each elected AN, together with the ANs that contributed to its election, constitute a community, which is identified by the profile of the elected node.

5 Structured Layer

GROUP enables the creation, in a self-emerging, distributed way, of communities made of devices characterized by similar profiles, namely "communities" of similar data streams. However, the protocol does not provide any support for indexing such communities. To overcome this limitation, we introduce a further layer to our architecture. The idea is to provide a distributed index based on a DHT specifically instrumented to perform approximate matches between a query and a community profile. The approximate search is obtained by exploiting a Locality Sensitive Hash (LSH) approach. This approach allows to find the community of data streams that is the closest one to that provided by means of a simulacrum. In fact, traditional DHTs are very efficient to support the search for exact uni-dimensional data, but they are not conceived for supporting approximate searches. The idea for achieving a support for approximated multi-attribute searches on DHTs has been initially proposed by Zhu [18]. The approach consists in applying the Locality-Sensitive Hashing (LSH) method [22], Specifically, a family of hash functions $\mathcal{H} \in \mathbb{R}^d$ is locality-sensitive if, given a random hash function $h \in \mathcal{H}$, for any pair of points a, $b \in \mathbb{R}^d$ and a distance threshold r, we have:

- if $\|a - b\| \leq r$ then $Pr[h(a) = h(b)] \geq p_1$
- if $\|a - b\| \geq r$ then $Pr[h(a) = h(b)] \leq p_2$

In other words, fixed $p_1 > p_2$ the hash function allows to map with high probability a and b in the same bucket if they are very close (according to a given threshold r) or in different buckets if they are quite different. A detailed description of LSH can be found in the paper of Antoni and Indyk [23]. In this paper we exploit LSH as a mechanism for supporting efficient approximated searches in DHTs. In particular, for each profile we create n different indices, which are used to register a profile in a DHT. A submitted query is first indexed with the same LSH method. Then the community representatives' profiles registered under the same indices are retrieved and compared against the query in order to carry out the most similar representatives. Finally, such representatives forward the query to the related community of devices that likely manipulated a data stream close to the one represented by the simulacrum provided as a query. In order to exploit the potentials of this indexing mechanism, we test this structured layer with the two different types of profile representations described in Sect. 3.

All of them are built starting from the attributes collections characterizing the profile of a node. Clearly, to compare two profiles (and queries against profiles), proper similarity functions must be used. For the Weighted Attribute Vectors profile model we use the following function:

$$SIM_V(P_1, P_2) = \frac{\sum\limits_{obj \in P_1 \cap P_2} min\,[W_1(obj), W_2(obj)]}{max(|P_1|, |P_2|)} \tag{1}$$

where P_1 and P_2 are the two profiles to compare, and $W_1(obj), W_2(obj)$ are the weights associated to *obj* within P_1 and P_2, respectively. Like the Jaccard similarity measure, this similarity is computed only on the intersection of the attributes shared by P_1 and P_2. For each of them the minimum weight is considered. The sum of all those values is weighted with the size of the largest profile. This is done in order to avoid having a high similarity degree even in case of a profile is completely or largely contained in the other, or even when it represents only a small subset of other profiles. In order to compare matrix-based profiles, i.e. when Adjacency Matrices is used, the previous formula is changed in:

$$SIM_M(P_1, P_2) = \frac{\sum\limits_{obj \in P_1 \cap P_2} [min\,(W_1(obj), W_2(obj)) \cdot \delta_{Rel}(obj)]}{max(|P_1|, |P_2|)} \tag{2}$$

where

$$\delta_{Rel}(obj) = \frac{\sum\limits_{obj' \in P_1 \cap P_2} min\,(Rel_1(obj, obj'), Rel_2(obj, obj'))}{\max\limits_{i=1,2} |\, \{obj' \in P_1 \cap P_2 | \exists Rel_i(obj, obj')\} \,|}$$

In such a case, in addition to the two profiles sizes and attribute weights, we exploit the function $\delta_{Rel}(\cdot)$. It measures the degree of relationship of each attribute of the Adjacency Matrices-based profile, with the other ones. More precisely, given an object *obj*, we consider only the attributes that are in the P_1 and P_2 intersection. For each attribute we consider the sum over the minimum relevance weights existing in the two profiles. The relevance with an object obj' is given by the function $Rel(\cdot, \cdot)$. This sum is weighted with the maximum size of the set of objects having a relation with *obj*. Note that using an adjacency matrix, this set has the same size on both profiles, because all objects are considered to have a relation, even when they have a value equals to 0. In order to analyse the advantages deriving by the usage of all profile models, we performed a theoretical comparison between two different solutions, also comparing the LSH approach against a naive solution that would work by indexing, storing and retrieving every attribute of each profiles.

Theoretical Analysis. Table 1 shows the theoretical costs computed considering the LSH indexing approaches when applied to index node profiles expressed according to *Weighted Attribute Vector* and *Adjacency Matrix* profiles. Such costs are computed as function of the number of profiles' attributes, namely a cost $O(n)$ means n times the amount of memory required to store (or transfer) an attribute. In our analysis $|P|$ indicates the number of attributes composing a profile. X indicates the number of peers composing the DHT network, *Com* the total number of registered communities (i.e. groups of similar data streams). k is the maximum number of profiles returned by a node of the DHT when it resolves a query. R is the number of accesses performed to update the DHT when a community profile changes. Each access removes a copy of a community profile at a certain key (corresponding to a profile attribute) that is no longer contained in the community profile. n indicates the number of LSH identifiers associated with each profile.

Table 1. Complexity analysis for indexing profiles.

Operation	Profile	LSH cost	Naive cost
Query	A.M.	$O(n \cdot \frac{\|P^2\|}{2} \cdot \log(X))$	$O(\frac{\|P^3\|}{2} \cdot \log(X))$
	T.V.	$O(n \cdot \|P\| \cdot \log(X))$	$O(\|P\|^2 \cdot \log(X))$
Query resolution	A.M.	$O(k \cdot \frac{\|P\|^2}{2} \cdot n)$	$O(k \cdot \frac{\|P\|^3}{2})$
	T.V.	$O(k \cdot \|P\| \cdot n)$	$O(k \cdot \|P^2\|)$
Community insertion	A.M.	$O(n \cdot \frac{\|P\|^2}{2} \cdot \log(X))$	$O(\frac{\|P\|^3}{2} \cdot \log(X))$
	T.V.	$O(n \cdot \|P\| \cdot \log(X))$	$O(\|P\|^2 \cdot \log(X))$
Profile update	A.M.	$O((n \cdot \frac{\|P\|^2}{2} + R) \cdot \log(X))$	$O((\frac{\|P\|^3}{2} + R) \cdot \log(X))$
	T.V.	$O(n \cdot \|P\| + R) \cdot \log(X))$	$O((\|P\|^2 + R) \cdot \log(X))$
Descriptor removal	A.M.	$O(n \cdot \log(X))$	$O(\|P\| \cdot \log(X))$
	T.V.	$O(n \cdot \log(X))$	$O(\|P\| \cdot \log(X))$
Index size	A.M.	$O(n \cdot \frac{\|P\|^2}{2} \cdot Com)$	$O(\frac{\|P\|^3}{2} \cdot Com)$
	T.V.	$O(n \cdot \|P\| \cdot Com)$	$O(\|P\|^2 \cdot Com)$

Weighted Attribute Vector model. Following the **naive** approach, searching for a profile to requires to send to the DHT a request for each profile's attribute. Thus, the generation of $O(|P|^2 \cdot \log(X))$ messages. This derives from the DHT logarithmic routing approach: for each attribute a profile copy is transferred to a logarithmic subset of the set of nodes realising the DHT. Each queried node answers by sending a message of $O(k \cdot |P|)$ elements to the k communities with a profile that is similar to the received query. As a consequence the total amount of messages exchanged is $O(k \cdot |P^2|)$. The creation of the distributed index requires, for each community, to store a copy of its profile, for each profile's attribute. This leads to $O(|P|^2 \cdot Com)$ messages. When a community profile changes, the index is updated, $|P|$ copies of the new profile are sent, one for each profile's attribute. Moreover, R additional notifications are sent, one for each attribute that is no longer part of the community profile, aimed at removing old community's profiles. As a consequence, the number of exchanged messages is equal to $O(|P|^2 \cdot \log(X) + R \cdot \log(X))$. Storing a new profile requires to sent $|P|$ copies of that profile, one for each profile's attribute. Consequently, the total amount of generated messages is equal to $O(|P|^2 \cdot \log(X))$. Removing a community profile requires to send $|P|$ messages, which correspond to the number of profile's attributes that equals to $O(|P| \cdot \log(X))$; Using the **LSH** model a community search requires to generate a query for each one of the n LSH identifiers computed for a peer profile. As a consequence the generated number of messages is independent by the number of attributes within a community's profile. Since any message contains a community profile, the total number of generated messages is equal to $O(n \cdot |P| \cdot \log(X))$. Like the naive solution, to answer a query each peer sends a message having a maximum size equal to $(k \cdot |P|)$, but only n nodes of the DHT are involved. This leads to a total amount of generated messages of $O(n \cdot k \cdot |P|)$. The number of profile copies stored along the distributed index is equal to the number of the n LSH identifiers associated to a community's profile. This implies that the total number of generated messages for storing the whole distributed index is equal to $O(n \cdot |P| \cdot Com)$. When a community profile is updated, the DHT is requested to store n copies of the new profile, leading to an equivalent number of profiles to transfer, and, in the worst case, other n requests of profile removal are generated for deleting the old profile. However, due to the LSH properties the number of identifiers exchanged between the old and new profile is less than n on average. Then, the overall number of generated messages for updating a community profile is $O(n \cdot |P| \cdot \log(X) + n \cdot \log(X))$. The removal of a community descriptor requires to send n messages that generate a number of exchanged messages equal to $O(n \cdot \log(X))$.

Attribute Adjacency Matrix model. When a profiles is structured as an adjacency matrix, its behaviour in terms of complexity, for the various operations, is pretty similar to the Weighted Attribute Term Vector model. The only notable difference is on the amount of information required to represent the profile. In this case it goes from $|P|$ to $\frac{|P|^2}{2}$. As a consequence, almost all the complexities are scaled of a factor $\frac{|P|}{2}$, with the only exception on the removal of a descriptor, that is not directly proportional to the profile size but on the number of attributes.

6 Evaluation

The focus of our solution is on enabling approximate queries over textual data (coming from data streams) in a distributed system based on IoT and Edge devices. To this end, we firstly, we measured the ability of our system to maintain high-quality community representatives (representatives of a collection of data streams) when the indexed data changes. Figure 1 shows the average similarity of community members with the selected representatives, and with the other members of the same community. This experiment has been conducted by varying the actual composition of the information extracted by the stream processing devices starting from the simulation cycle #50. Every cycle we changed the 5% of the information content of a randomly selected set representing the 2% of the nodes. As can be observed, the similarity of nodes with their representative is essentially not affected by the changes. Thus the system is able to adaptively react to changes. Then, we analyse the ability of our system to

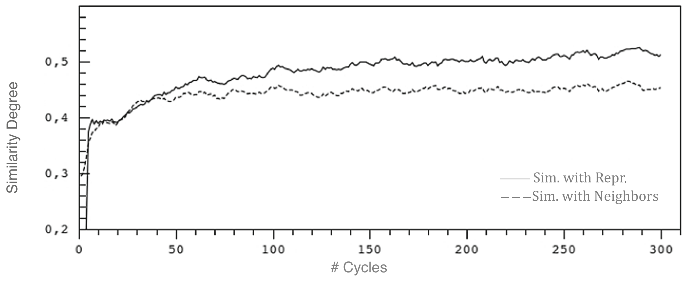
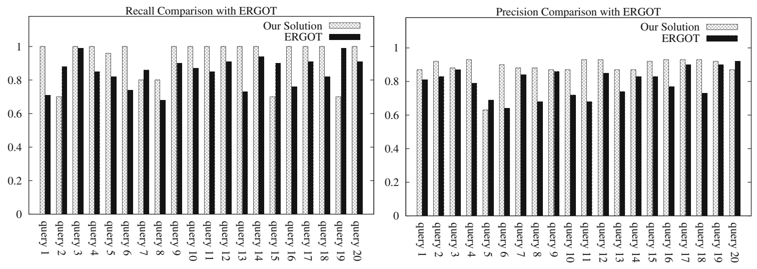


Fig. 1. Dynamic behaviour: internal homogeneity of communities

efficiently resolve textual queries in a distributed fashion. This evaluation is made in comparison against ERGOT [14]. ERGOT is a DHT-based Semantic Overlay Network aimed at service discovery structured on two layers (structured and unstructured). The main difference with our solution relies on the actual viewpoint. They use a DHT-based structured layer as lower-layer and a semantic-based unstructured network as higher-layer. To conduct an effective comparison we directly contacted the authors of ERGOT that provided us the dataset they used in their evaluation, the source code of their proposed solution and the complete set of information about the configuration of their testing environment. The dataset has been built by exploiting the WordNet ontology and the WordNet domain [24]. It consists of 200 domains labels organized in a hierarchical structure that categorizes WordNet synsets into domains. The content of this dataset has been used for generating textual descriptions, which has been assigned to profiles according to a Zipf distribution. The evaluation has been focused on the ability of retrieving relevant profiles given an input query.



(a) Recall (b) Precision

Fig. 2. Comparison against ERGOT

A result has been considered as *relevant* if its degree of similarity with the query is greater than 0.5. For achieving a fair trial, in our evaluation we compared the results obtained by ERGOT and NOA-AID, using the set of 20 queries presented in the original paper of ERGOT. Figure 2a shows the comparison between the results achieved by ERGOT and the ones provided by NOA-AID in terms of Recall. The Recall value for a given query has been computed as the ratio of the number of relevant profiles obtained on the total number of relevant results existing in the system. It can be noticed how our solution provides better results in almost all the submitted queries. Figure 2b shows the results obtained for the Precision metric by the two approaches. Precision has been computed as the ratio of the number of relevant peer profiles retrieved on the total number of profiles returned. Also in this case using our solution provide a clear advantage with respect to ERGOT in almost all cases. To validate the efficiency of our proposed system in providing access to the information sought by a requester (simulacrum describing the data stream), while minimizing the amount of data transferred through the network, we measure the actual cost associated with the LSH-based indexing and query resolution techniques. Their performance are strongly relevant for resource-constrained devices, like the ones we are focusing in this paper. Table 2 reports both the amount of data needed for storing the whole index of the communities as well as the communication costs associated with the query resolution process. Results are compared against the Naive solution. As can be observed, the experimental results validate the expected theoretical behaviour, following from the evaluation presented in Sect. 5.

Table 2. Comparison of the theoretical load with the actually measured one.

	Parameters	Naive	NOA-AID	Exp. Gain	Meas. Gain
Index size					
Term vector	n = 15; P = 300	3064	148	20	20.6
	n = 20; P = 300	3064	190	15	16.11
Adj. Matrix	n = 15; P = 300	257557	12728	20	20.2
	n = 20; P = 300	257557	16970	15	15.2
Query resolution					
Term vector	n = 15; P = 300	22165	1210	20	18.3
	n = 20; P = 300	22165	1544	15	14.35
Adj. Matrix	n = 15; P = 300	1572155	78684	20	19.98
	n = 20; P = 300	1572155	104810	15	15

7 Conclusions

In this paper, we propose NOA-AID, a network architecture aimed at providing a flexible query-by-example indexing and discovery mechanism targeting stream processing devices belonging to a highly dynamic and distributed environments. It is based on two overlay networks. At the lower level lies an unstructured, epidemic-based, network able to autonomously adapt and self-organize, aimed at grouping stream processing devices, collecting an heterogeneous set of information, into communities. The higher network layer indexes such communities and provides a query-by-example solution easing their discovery. We provided both a theoretical as well as a experimental evaluation of the approach showing its effectiveness and efficiency.

References

1. Cai, M., Frank, M., Chen, J., Szekely, P.: MAAN: a multi-attribute addressable network for grid information services. J. Grid Comput. **2**(1), 3–14 (2004)
2. Chang, R.S., Hu, M.S.: A resource discovery tree using bitmap for grids. Future Gener. Comput. Syst. **26**, 29–37 (2010)
3. Marzolla, M., Mordacchini, M., Orlando, S.: A P2P resource discovery system based on a forest of trees. In: 17th International Workshop on Database and Expert Systems Applications (DEXA 2006), pp. 261–265 (2006)
4. Mordacchini, M., Ricci, L., Ferrucci, L., Albano, M., Baraglia, R.: Hivory: range queries on hierarchical voronoi overlays. In: IEEE Tenth International Conference on Peer-to-Peer Computing (P2P2010). IEEE, pp. 1–10 (2010)
5. Gennaro, C., Mordacchini, M., Orlando, S., Rabitti, F.: Mroute: a peer-to-peer routing index for similarity search in metric spaces. In: Proceedings of the 5th International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2007), pp. 1–12 (2007)
6. Bruno, R., Conti, M., Mordacchini, M., Passarella, A.: An analytical model for content dissemination in opportunistic networks using cognitive heuristics. In: Proceedings of the 15th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems. ACM, pp. 61–68 (2012)
7. Liu, L., Antonopoulos, N., Mackin, S., Xu, J., Russell, D.: Efficient resource discovery in self-organized unstructured peer-to-peer networks. Concurr. Comput. Pract. Exp. **21**, 159–183 (2009)
8. Ruffo, G., Schifanella, R.: A peer-to-peer recommender system based on spontaneous affinities. ACM Trans. Internet Technol. **9**, 4:1–4:34 (2009)
9. Baraglia, R., Dazzi, P., Guidi, B., Ricci, L.: Godel: Delaunay overlays in P2P networks via gossip. In: IEEE 12th International Conference on Peer-to-Peer Computing (P2P). IEEE, pp. 1–12 (2012)
10. Mordacchini, M., Dazzi, P., Tolomei, G., Baraglia, R., Silvestri, F., Orlando, S.: Challenges in designing an interest-based distributed aggregation of users in P2P systems. In: 2009 International Conference on Ultra Modern Telecommunications & Workshops, ICUMT 2009. IEEE, pp. 1–8 (2009)
11. Danelutto, M., Dazzi, P., et al.: A Java/Jini framework supporting stream parallel computations. In: PARCO, pp. 681–688 (2005)
12. Lulli, A., Ricci, L., Carlini, E., Dazzi, P., Lucchese, C.: Cracker: crumbling large graphs into connected components. In: 2015 IEEE Symposium on Computers and Communication (ISCC). IEEE, pp. 574–581 (2015)
13. Falchi, F., Gennaro, C., Zezula, P.: Nearest neighbor search in metric spaces through content-addressable networks. Inf. Process. Manag. **43**(3), 665–683 (2007)
14. Pirrò, G., Talia, D., Trunfio, P.: A DHT-based semantic overlay network for service discovery. Future Gener. Comput. Syst. **28**(4), 689–707 (2012)

15. Guerraoui, R., Sidath, B., Kermarrec, A., Fessant, F.L., Huguenin, K., Rivière, E.: GosSkip, an efficient, fault-tolerant and self organizing overlay using gossip-based construction and skip-lists principles. In: Sixth IEEE International Conference on Peer-to Peer Computing, 2006 Ratnasamy, pp. 12–22 (2001)
16. Crespo, A., Garcia-Molina, H.: Semantic overlay networks for P2P systems. In: Moro, G., Bergamaschi, S., Aberer, K. (eds.) AP2PC 2004. LNCS (LNAI), vol. 3601, pp. 1–13. Springer, Heidelberg (2005). https://doi.org/10.1007/11574781_1
17. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston (2005)
18. Zhu, Y., Hu, Y.: Efficient semantic search on DHT overlays. J. Parallel Distrib. Comput. **67**(5), 604–616 (2007)
19. Baraglia, R., Dazzi, P., Mordacchini, M., Ricci, L.: A peer-to-peer recommender system for self-emerging user communities based on gossip overlays. J. Comput. Syst. Sci. **79**(2), 291–308 (2013)
20. Baraglia, R., Dazzi, P., Mordacchini, M., Ricci, L., Alessi, L.: GROUP: a gossip based building community protocol. In: Balandin, S., Koucheryavy, Y., Hu, H. (eds.) NEW2AN/ruSMART -2011. LNCS, vol. 6869, pp. 496–507. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22875-9_45
21. Carlini, E., Dazzi, P., Mordacchini, M., Ricci, L.: Toward community-driven interest management for distributed virtual environment. In: an Mey, D., et al. (eds.) Euro-Par 2013. LNCS, vol. 8374, pp. 363–373. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54420-0_36
22. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: Proceedings of the International Confernece on Very Large Data Bases, pp. 518–529 (1999)
23. Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Commun. ACM **51**(1), 117–122 (2008)
24. Bentivogli, L., Forner, P., Magnini, B., Pianta, E.: Revising wordnet domains hierarchy: semantics, coverage, and balancing. In: Proceedings of COLING 2004 Workshop on Multilingual Linguistic Resources, pp. 101–108 (2004)