Engineering Democratization in Internet of Things Data Analytics

Evangelos Pournaras, Jovan Nikolić, Aleš Omerzel, Dirk Helbing
Professorship of Computational Social Science
ETH Zurich
Zurich, Switzerland
Email: {epournaras,jnikolic,omerzela,dhelbing}@ethz.ch

***Abstract*—The pervasiveness of Internet of Things devices in techno-socio-economic domains such as Smart Cities and Smart Grids results in a massive scale of data about our society. Decision-making by system operators or policy-makers requires a sophisticated understanding of these data with real-time data analytics methods. However, common data analytics methods often serve exclusively corporate and commercial interests and result in privacy-intrusion, surveillance, profiling and discriminatory actions. This paper illustrates an alternative data analytics approach that relies on participatory citizens to contribute Internet of Things data and crowdsourced computational resources in order to compute aggregation functions in a collective fashion. This democratization calls for a fully decentralized and privacy-preserving system design with which a local data management mechanism implemented in smart phones can guarantee highly accurate computations under highly dynamic data streams. Experimental evaluation with real-world Smart Grid data illustrates the performance trade-offs and shows how they can be managed in an automated and empirical way using decision trees.**

I. Introduction

The massive expansion of data collection sources from Internet of Things devices such as pervasive and ubiquitous sensors in the context of Smart Cities, wearables and smart phones [1] brings unprecedented opportunities to reason about our society based on data-driven empirical evidence. Internet of Things data analytics has turned out to be a tactical utility to turn tremendous scales of massive unstructured data to economic merchandises and services with implications in the societal, environmental, economic and political arena. Unavoidably, the question here is if practices of existing data analytics contribute to the sustainability of digital societies. Challenges such as privacy-intrusion, surveillance and discrimination data analytics algorithms using personal data [2] can put the cohesion of society into question.

A democratization of data analytics in the Internet of Things era can turn data analytics into a public good to tackle the aforementioned challenges [3]. This requires a shift from an exclusively commercial scope to a broader scope in which participating citizens contribute computational resources and data in a more ethical way than existing data collection practices do. To make this happen, a fundamentally different engineering design is required that goes beyond distributed models such as MapReduce [4], [5], [6] running in large and expensive computational machines centrally managed by stakeholders with entirely commercial interests.

This paper illustrates a fully decentralized privacy-preserving data analytics system for the Internet of Things. Given the well documented challenge and complexity to design, deploy and manage such scalable systems [7], [8], this paper focuses on the problem of dynamic decentralized aggregation of Internet of Things data: citizens using pervasive and ubiquitous devices act as data suppliers and consumers generating streams of real-time privacy-sensitive data, for instance, sensor data from mobile devices or power consumption data from residential smart meters. Citizens are incentivized to contribute some of these dynamically changing data they collect to get collective information in return, such as traffic congestion information or status updates of the Smart Grid reliability [9]. This information is the output of aggregation functions, such as SUMMATION, AVERAGE, MAXIMUM, MINIMUM, TOP-K, computed via peer-to-peer interactions within a decentralized network of crowdsourced computational resources. The focus of this paper is the decentralized and privacy-preserving computation of aggregation functions under highly rapid changes of input data from the Internet of Things that challenges the cost-effectiveness of any decentralized data management system.

This paper contributes a local data management mechanism, implemented as an app on smart phones for proof of concept, that can guarantee high aggregation accuracy under highly entropic data streams that challenge the continuous update in the estimation of aggregation functions. Experimental evaluation with real-world data from the domain of Smart Grids contributes novel quantitative findings about performance trade-offs, such as privacy vs. accuracy vs. communication cost, which confirm the feasibility of the proposed data analytics approach. This paper also illustrates a data-driven decision support method based on decision trees for regulating the performance trade-offs in an automated fashion by exclusively using high-level parameters.

The remainder of this paper is organized as follows: Section II introduces a data management model for Internet of Things devices that can perform a fully decentralized aggregation of dynamic sensor data. Section III illustrates the experimental evaluation. Section IV compares the work of this paper with related work. Finally, Section V concludes this paper and outlines future work.

II. Decentralized Aggregation in the Internet of Things

This paper studies the applicability of a fully decentralized data analytics system for the Internet of Things. Pervasive and ubiquitous devices participate as data suppliers and consumers.

Data suppliers are sensors that locally generate a stream of real-time data, while *data consumers* are devices that require access to collective information computed by aggregation and advanced analytics over the sensor data. This paper focuses on aggregation functions such as AVERAGE, SUMMATION, MAXIMUM, MINIMUM, STANDARD DEVIATION and TOP-K that are typically challenging to compute in a fully decentralized networked environment. The computational problem is also know as 'in-network aggregation'.

Data suppliers and consumers interact with a fully decentralized aggregation network built by the computational resources of participatory citizens. Such a network can be formed by the citizens themselves as done in the case of the diaspora decentralized social network that consists of several distributed servers referred to as pods maintained by the user community [10], [11]. Moreover, Do-It-Yourself decentralized networking [12], [13] can be applied in "offline" local area networks using wireless technology that provides coverage at a level of a neighborhood up to the level of a whole city, as in the case of LoRaWAN [14]. There is also the option to bootstrap such decentralized networks for Internet of Things data analytics with peer-to-peer clients such as the ones developed for Bittorrent [15].

For the purpose of illustration and analysis, the following simple, yet challenging, distributed scenario is studied: every data supplier comes along with a data consumer. Each pair of them is connected to a different node in the aggregation network. The data supplier and consumer for each node are assumed to be located in a citizen's device such as a smartphone or a home gateway that, for example, sends and receives power consumption data. Figure 1 illustrates the studied model.

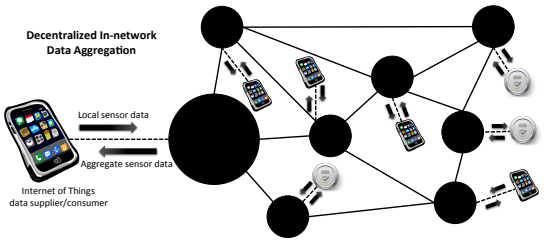


Figure 1. Decentralized in-network data aggregation for the Internet of Things. Pervasive and ubiquitous devices locally generate data that are sent to a node in the aggregation network with which they are connected. Aggregation functions for the same type of data are computed and the aggregate results are sent back to the devices.

A. Decentralized data aggregation

This paper employs DIAS, the *Dynamic Intelligent Aggregation Service* for the computation of aggregation functions [16]. DIAS can perform decentralized privacy-preserving data analytics as it relies on local computations, peer-to-peer interactions and hashed information. It is applicable in the context of the Internet of Things given that it can manage dynamic input data streams from sensors as it is shown in this paper. Although input sensor data rapidly change and introduce errors to the actual true values of the computed aggregation functions, DIAS is designed to perform automated and continuous self-correcting operations that maintain a high accuracy in the estimations of the aggregation functions. Moreover, DIAS can compute a wide spectrum of aggregation functions such as AVERAGE, SUMMATION, MAXIMUM, MINIMUM and STANDARD DEVIATION without any change in its core mechanism.

Figure 2 illustrates the main components and interactions of the DIAS aggregation service. Each node i of the DIAS network can contain a *disseminator* d_i and/or an *aggregator* a_i. The disseminator is the agent with which the data supplier is connected to, whereas the aggregator is the agent with which the data consumer is connected to. Disseminators discover aggregators in the network to which they send their local sensor data. Discovery is performed via a fully decentralized gossiping protocol, the *peer sampling service* [17] in which aggregators publish themselves and disseminators sample aggregators published. Disseminators classify and cache aggregators sampled from the peer sampling service in a pool of limited size and spread the local sensor data in the network periodically by pushing them to remote aggregators from the pool. Aggregators collect the input data for the computation of the aggregation functions. Disseminators pull back information about the outcome of the performed aggregation. Each bilateral peer-to-peer interaction between a disseminator and an aggregator is referred to as an *aggregation session*. There are multiple parallel aggregation sessions performed during the runtime of DIAS.

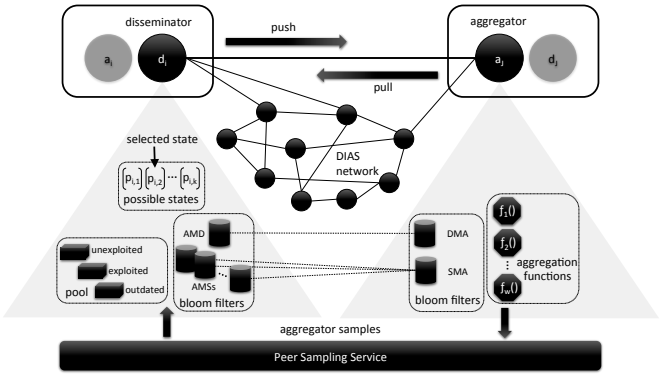


Figure 2. Overview of the DIAS main components and interactions. A disseminator d_i contains the possible states and the selected state. It has a memory system based on simple and counting bloom filters [16] that are used to classify aggregators as exploited, unexploited or outdated. The aggregators are sampled from a decentralized gossiping protocol, the peer sampling service [17]. An aggregator a_j shares part of the distributed memory system that is based on simple and counting bloom filters. It computes several aggregation functions, whose results are made available to data consumers of Internet of Things applications. The remote aggregators and disseminators in the DIAS network perform aggregation sessions, each of which consists of two messages exchanged in a push-pull peer-to-peer fashion.

The two main outcomes of each aggregation session are the following: (i) *exploitation*, during which a first contact with a certain disseminator is performed and therefore new input data are counted in the aggregation functions, and (ii) *update*, during which a contact with a certain disseminator is repeated to encounter changes occurred in the earlier data received. The updates are the means to apply the self-corrective operations on the computation of the aggregation functions. Determining these two outcomes requires that DIAS agents have some memory with which they can distinguish (i) the *interactions* and (ii) the *data received*. A detailed illustration of the intelligent memory system of DIAS is beyond the scope of this paper and explained in more detail in earlier

work [16]. However it must be underlined, that the memory system relies on the probabilistic data structures of simple and counting bloom filters, whose consistency is maintained in a fully distributed way using exclusively the interactions of an aggregation session. A remote pair of simple bloom filters[1] keeps information about the mutual interactions of disseminators-aggregators, and a set of other remote counting bloom filters[2] keeps information about the mutual data exchanged for aggregation.

The main data abstraction model of DIAS is the concept of possible states. Each disseminator i in the network has a sequence of k possible states $P_i = (p_{i,u})_{u=1}^{k}$. Moreover, each disseminator i has one and only one selected state $p_{i,s} \in P_i$, which is the one that is counted as input in the aggregation functions. A disseminator can change its selected state for another possible state any time during its operation. The actual value of an aggregation function is computed for n nodes in the evaluation as $f(p_{1,s}, ..., p_{n,s})$, whereas the aggregate estimated by DIAS is notated as $\overline{f}(p_{1,s}, ..., p_{n,s})$.

Algorithm 1 illustrates the active thread of a disseminator d_i executed periodically with period T (line 2). The disseminator samples (line 3) from the peer sampling service a set of aggregators A that classifies (line 4) in the aggregation pool as (i) *exploited*, (ii) *unexploited* or (iii) *outdated*. Exploited are the aggregators with which the disseminator has already interacted and aggregated their current selected state, in contrast to the unexploited ones with which the disseminator has not interacted. The disseminator has interacted with the outdated aggregators, however, they have aggregated a different possible state than the current selected one. These distinctions are made by querying the bloom filter memory system. An aggregator a_j is selected from the pool to initiate an aggregation session (line 5). Selection is made based on a strategy s that gives priority to either unexploited or outdated aggregators. These two priority schemes are the (i) *exploitation* and (ii) *update* strategies. A memory report m_{d} is formed by querying the bloom filters for the selected aggregator (line 6) that is wrapped into a 'push' message together with the possible states and the selected state. The message is sent to the selected aggregator a_j (line 7).

Algorithm 1 Active thread of disseminator d_i.

Require: Strategy s
1: **loop**
2: wait(T)
3: A=service.sample()
4: pool.classify(A)
5: a_j=pool.select(s)
6: m_{d}=memory.recall(a_j)
7: send('push',d_i,m_{d},P_i,$p_{i,s}$) to a_j
8: **end loop**

Algorithm 2 illustrates the passive thread of the aggregator a_j listening to 'push' messages. When such a message is received, the aggregator recalls from its bloom filter memory information about the disseminator d_i to validate[3] whether the classification to unexploited or outdated performed by the disseminator d_i is correct (line 1). The outcome of this memory recall is the aggregator report r_{a}. Then the aggregation process starts by updating the memory system (line 2) and computing the aggregation functions (line 3). Finally, a 'pull' message is sent back to the disseminator d_i containing the aggregator report r_{a} (line 4).

Algorithm 2 Passive thread of aggregator a_j.

Require: 'push',d_i,m_{d},P_i,$p_{i,s}$
1: r_{a}=memory.recall(d_i,m_{d})
2: memory.memorize(m_{d})
3: aggregate($p_{i,s}$,P_i,r_{a})
4: send('pull',a_j,r_{a}) to d_i

The 'pull' message is processed by the passive thread of the disseminator d_i illustrated in Algorithm 3. The processing involves the update of the memory system (line 1) for the consistency of the future generated disseminator reports and the classifications of aggregators in the aggregation pool.

Algorithm 3 Passive thread of disseminator d_i.

Require: 'pull',a_j,r_{a}
1: memory.memorize(r_{a})

Figure 3 visualizes the operation of DIAS with execution period $T = 1$ second that is the duration of an epoch measuring the DIAS runtime. The execution period of the peer sampling service is $T/4$. The extreme scenario in which every node of the network has an aggregator and a disseminator is studied. After a bootstrapping period of 13 epochs, the connections of the peer sampling service are established as shown in Figure 3a-b. The aggregation pool of disseminators is filled with several unexploited aggregators and all nodes start establishing several aggregation sessions as shown in Figure 3c-f. Figure 3g-l shows that after the first iterations, the number of aggregation sessions decreases as fewer and fewer unexploited aggregators fill the aggregation pool. Disseminators do not change their selected state and the network size remains constant. The estimation of the aggregation functions converges to the actual values.

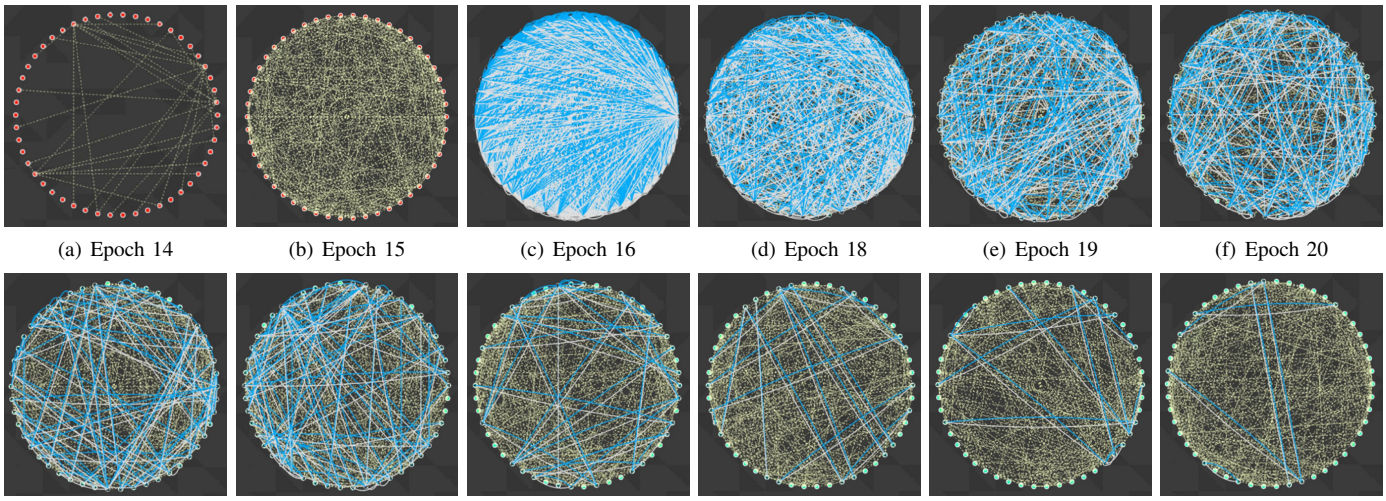
B. Data management for the Internet of Things

Sensors generate streams of real-time data referred to in this paper as *raw data*. For example, a smart meter may measure the energy consumption of a household generating a large number of positive real numbers. These numbers may be privacy-sensitive. They can also significantly and rapidly vary over time, which can turn the use of raw data as possible states infeasible in DIAS. The DIAS nodes cannot handle and even know all the possible values generated by a sensor. Moreover, the selected states from the raw data change in real-time as data are collected. This turns the self-correcting actions of DIAS on the aggregation functions infeasible as well. In addition, raw data are sensitive and the users of the devices may be unwilling to permit these data leaving their devices as a measure to preserve their privacy. It is concluded that the privacy-sensitive, highly entropic and dynamic sensor data generated from Internet of Things devices is the challenging factor for performing truly decentralized data analytics, in

[1]These are the Aggregator Memberships in Disseminator (AMD) and the Disseminator Memberships in Aggregator (DMA).

[2]These are the Aggregator Memberships in Selected states (AMSs) and the Selected state Memberships in Aggregates (SMA).

[3]This validation performs a consistency check for false positives that bloom filters can generate.



(a) Epoch 14 (b) Epoch 15 (c) Epoch 16 (d) Epoch 18 (e) Epoch 19 (f) Epoch 20

(g) Epoch 21 (h) Epoch 22 (i) Epoch 23 (j) Epoch 24 (k) Epoch 25 (l) Epoch 26

Figure 3. Visualization of DIAS with 50 nodes, each having a disseminator and an aggregator. The dashed yellow lines indicate the connections established by the peer sampling service, whereas the white and blue solid lines indicate the push-pull peer-to-peer messages of the aggregation sessions. The nodes are colored red at the very beginning, indicating maximal errors in the computation of the aggregation functions. As more aggregation sessions are performed, the nodes turn to green, indicating maximal accuracy by estimating the actual values of the aggregation functions correctly. DIAS eliminates the communication cost as the accuracy increases and the disseminators communicate with all available unexploited and outdated aggregators.

DIAS and elsewhere. This section illustrates a solution to address this challenge.

The proposed solution relies on a summarization unit that transforms the raw data into a stream of selected states chosen from a limited number of k possible states. This transformation turns the raw data to the *summarized data*. The summarized data are representative values of the raw data, and they can be locally computed using data mining and machine learning techniques applied on historical raw data [18]. For example, instead of sending to DIAS the exact power consumption readings from a smart sensor, a stream of three possible values can be sent that represent the low, medium and high power consumption profiles of a household. These profiles can be extracted by clustering [19] historical raw data with an algorithm such as k-means and using the centroid of the clusters as possible states. This training process can be repeated at much larger intervals than the data collection, e.g. daily or weekly. A future raw data record r_i generated by a sensor can be tested in which cluster it belongs to by using a distance measure such as the Euclidean distance. In this case, the selected state is determined as follows:

$$s = \arg\min_{u=1}^{k}(|r_i - p_{i,u}|), \tag{1}$$

where s is the index of the selected state $p_{i,s} \in P_i$ and r_i is the most recent raw data record generated from a sensor. The summarized data have a lower information content than the raw data and therefore a level of privacy is introduced.

The number of possible states is selected as a system parameter controlling the storage and communication cost of DIAS and the privacy-preservation by looking at the number of possible states as the level of information reveal. The number of possible states can be also selected in an automated fashion via machine learning or data mining techniques. For example, the expectation maximization (EM) algorithm [19] indicates the number of clusters that best represents some given data.

Moreover, data suppliers may choose if they report every change of the selected state to the DIAS network. This is a way to control and regulate (i) the privacy-preservation, (ii) the communication cost between devices and DIAS nodes and (iii) the communication cost within DIAS. This is achieved with the *send factor* (SF) that determines a repeated time period[4] T_s in which the selected states are sent to DIAS. For example, a send factor of 3 means that the selected state is sent to DIAS every $3 * T$. Algorithm 4 illustrates how the send factor is applied in the summarization unit of Internet of Things devices.

Algorithm 4 Influencing the reported selected state in DIAS using the send factor.

Require: T_s, r_i, P_i
1: $t = 0$
2: **loop**
3: wait(T)
4: $t = t + 1$
5: **if** $t = T_s$ **then**
6: $s = \arg\min_{u=1}^{k}(|r_i - p_{i,u}|)$
7: send $p_{i,s}$
8: $t = 0$
9: **else**
10: skip sending $p_{i,s}$
11: **end if**
12: **end loop**

The implementation of such a summarization unit depends on the Internet of Things devices used and the application. In devices with very low processing and power capacity, the possible states can be preprogrammed and updated manually. An automated lightweight data analysis, e.g. frequencies of the generated sensor values, is feasible in devices

[4]This period should be usually equal or larger than the main execution period of DIAS.

with small processing units [20]. In devices such as smart phones and larger smart sensor modules, more options are available. Mobile phone frameworks such as Nervousnet [21] or JAM [22] can efficiently manage sensor data and locally perform lightweight data mining and machine learning to transform raw into summarized data on a daily or weekly basis as shown is Section III-A. There is also the option to deploy the summarization unit on the DIAS node. The raw data need to be transferred, given available bandwidth and security measures.

Figure 4 illustrates the data lifecycle in the proposed decentralized aggregation system for the Internet of Things. Data suppliers generate raw data that are transformed to summarized data with the summarization unit before they are used by the DIAS network for aggregation. Finally, the DIAS nodes send back aggregation data to data consumers.

Three types of relative errors are introduced within this data lifecycle: (i) *summarization error*, (ii) *DIAS error* and (iii) *overall error*. The summarization error is a result of compressing the raw data to the possible states and introducing the send factor that does not always send the selected state to DIAS. It is computed for an aggregation function as follows:

$$\epsilon_{\mathsf{s}} = \frac{|f(r_1, ..., r_n) - f(p_{1,s}, ..., p_{n,s})|}{|f(r_1, ..., r_n)|} \tag{2}$$

From a privacy perspective, this error should be maximized, however, the errors introduced at this stage unavoidably accumulate at the later stages. One relevant question studied in Section III is how these local summarization errors accumulated influence the global errors in the estimations of the aggregation functions at the last stage given that these errors may cancel out in aggregation. The DIAS error is influenced by the complexity of the input data and system parameters such as the communication frequency and the network size. It is computed as follows:

$$\epsilon_{\mathsf{d}} = \frac{|f(p_{1,s}, ..., p_{n,s}) - \overline{f}(p_{1,s}, ..., p_{n,s})|}{|f(p_{1,s}, ..., p_{n,s})|} \tag{3}$$

The influence of system parameters is studied in earlier work [16]. This paper studies the influence of the send factor in the DIAS error as a measure of the data complexity in aggregation. The overall error is the most critical one that should remain minimal and is measured between the raw data aggregated and the estimated aggregation data of DIAS:

$$\epsilon_{\mathsf{o}} = \frac{|f(r_1, ..., r_n) - \overline{f}(p_{1,s}, ..., p_{n,s})|}{|f(r_1, ..., r_n)|} \tag{4}$$

The goal of this paper is to study the trade-offs between the three errors and to illustrate how to creatively use the send factor to regulate these trade-offs.

III. Experimental Evaluation

The experimental evaluation has the following two objectives: (i) show the feasibility of computing possible states in Internet of Things devices such as smart phones and (ii) show the feasibility of performing fully decentralized aggregation of dynamic sensor data. These two contributions provide a proof-of-concept at both a device (local) and network (global) level.

The former objective is studied by implementing an Android app using the Nervousnet[5] framework [21] that runs the k-means clustering algorithm on smart phones. The centroids of the clusters are the actual possible states, the summarized data. The following historical sensor data are used as raw data: (i) ACCELEROMETER, (ii) LIGHT and (iii) ACCELEROMETER-LIGHT. The latter concerns a virtual sensor that combines the sensor values of the former two. Therefore, a two-dimensional clustering is performed in this case. A single-day sensor data are collected at four different frequency periods: (i) 60 sec, (ii) 30 sec, (iii) 10 sec and (iv) 1 sec. The performance is measured by the average execution time over 30 repetitions. The measurements are performed on three different phones:

Phone 1 Motorola XT1039 (Android 5.1)[6].
Phone 2 Samsung SM-G920F Galaxy S6 (Android 6.0.1)[7].
Phone 3 Samsung GT-19505 Galaxy S4 (Android 5.0.1)[8].

For the latter objective, a Smart Grid scenario is emulated using real-world data from the *Electricity Customer Behavior Trial* (ECBT) pilot project[9], which studies the electricity consumption of residential consumers in Ireland. The project ran during the period 2009-2010 with 6435 residential and small-medium enterprise consumers, from which 3000 residential consumers are used for the experiments. Consumption data are collected from smart meters every 30 minutes. The data of date 4.1.2009 are used for the experiments. The total records of raw data used in the experiments are 2 records/hour*24 hours=48 records.

The data of ECBT are treated as the raw data. The summarized data are derived by performing clustering with k-means, where $k = 5$, using the Weka library[10].

The centroid of the clusters are the possible states of DIAS and the selected state is chosen every 30 minutes using Equation 1. Several send factors are evaluated: (i) SF-1, corresponding to sending the selected state every 1 epoch, (ii) SF-2 corresponding to sending the selected state every 2 epochs, (iii) SF-4, (iv) SF-8, (v) SF-12, (vi) SF-16 and so on. The estimated aggregates under different send factors are compared to the ACTUAL aggregates computed by the selected states without skipping changes in the selected states.

DIAS is implemented[11] in Java and prototyped using the Protopeer toolkit [23]. A Protopeer implementation[12] of the peer sampling service is used as well. Both DIAS and

[5]Available at: https://github.com/nervousnet (last accessed: October 2016).

[6]Specification available at http://www.knowreviewtip.com/specs-price/moto-g-4g-lte-1st-gen-xt1039/ (last accessed: October 2016).

[7]Specification available at http://www.phonemore.com/phone/samsung-galaxy-s6-sm-g920f-32gb/2043 (last accessed: October 2016).

[8]Specification available at http://www.gsmarena.com/samsung_i9505_galaxy_s4-5371.php (last accessed: October 2016).

[9]Available at http://www.ucd.ie/issda/data/commissionforenergyregulationcer/ (last accessed: October 2017)

[10]Available at https://weka.wikispaces.com (last accessed: October 2016).

[11]An early implementation of DIAS can be accessed at https://github.com/epournaras/DIAS (last accessed: October 2016). A latest implementation including new features such as fault tolerance, new aggregation functions, deployment support capabilities and other is available upon request.

[12]Available at https://github.com/epournaras/PeerSamplingService (last accessed: October 2016).

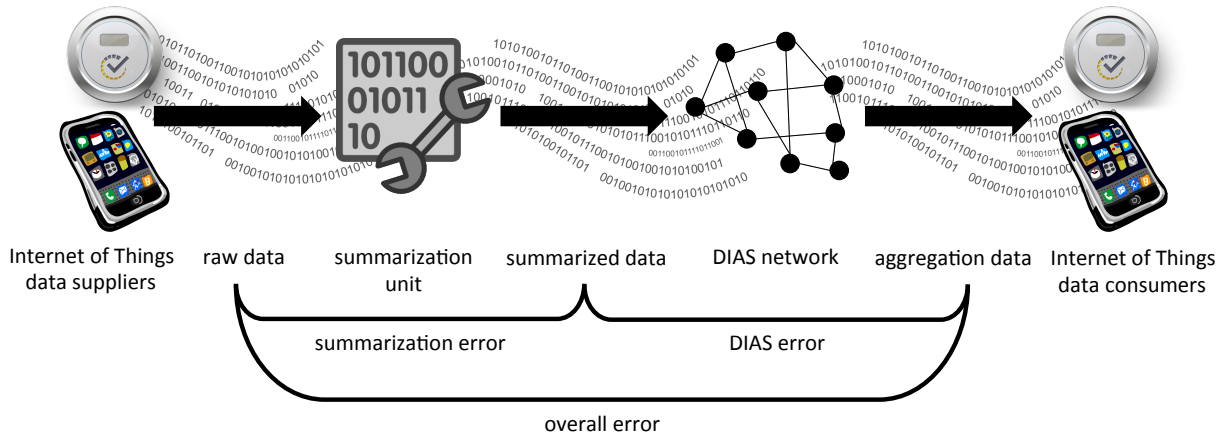


Figure 4. The data lifecycle starting from data suppliers and ending with data consumers in the Internet of Things. Raw data are transformed to summarized data resulting in a summarization error. Summarized data are used as input in the aggregation functions of DIAS to compute the aggregation data. The errors between the actual summarized data aggregated and the estimated aggregation data is the DIAS error. The errors between raw data aggregated and the estimated aggregation data in DIAS is the overall error.

peer sampling service are deployed in the Euler HPC cluster infrastructure of ETH Zurich. The epoch duration is selected as 30 minutes/14 DIAS executions=2.14 minutes (2.14/4=0.5 minutes for the peer sampling service). A high execution rate of DIAS improves convergence speed but also increases the communication overhead. It is a system parameter and should be selected based on the available resources of the infrastructure in which DIAS is deployed. Each of the 3000 nodes of DIAS are equipped with both a disseminator and an aggregator to test the most demanding scenario. A maximum of 40 aggregation sessions per epoch are initiated by each disseminator. Each experiment runs for 800 epochs in total, during which aggregation is performed in 14*48=672 epochs. The first 100 epochs are used for system bootstrapping. The results for the AVERAGE, SUMMATION and MAXIMUM aggregation functions are presented.

A. Local computation of possible states

Figure 5 illustrates the computational overhead involved in the computations of the possible states in three smart phones. The following observations can be made: Increasing the number of clusters from $k = 3$ to $k = 5$ increases the average computational overhead for each phone by 57.38%, 44.56% and 44.23% respectively.

Phone 2 with 2.1 GHz CPU has the lowest average computational overhead of 686.18 ms for $k = 3$ and 1061.38 ms for $k = 5$. In contrast, the average computational overhead increases in phone 1 with 1.2 GHz CPU to 1.59 sec and 3.11 sec respectively. For phone 3 with 1.9 GHz CPU the respective numbers are 2.78 sec and 5.51 sec.

In all cases, the number of possible states can be computed in a few seconds, which makes the proposed data management model feasible for pervasive devices such as smart phones.

B. Errors, privacy and communication cost

Figure 6 illustrates the summarized data and summarization errors for different send factors and aggregation functions. As it can be seen in Figure 6a and 6c, SF-1 does not vary significantly from the ACTUAL in AVERAGE and SUMMATION but it does vary significantly from SF-4 and SF-12. In MAXIMUM, the level of relative errors and the difference between the send factors are lower than the other aggregation functions. The errors are maximized when the level of energy consumption changes rapidly, e.g. leaving and returning home.

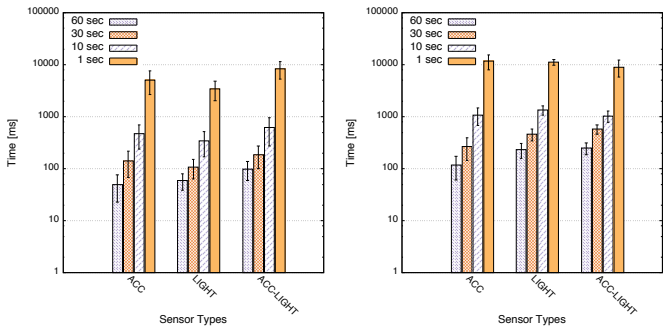
Figure 7 shows the aggregation data and the overall errors for different send factors and aggregation functions. With a careful look and cross comparison with Figure 6, one can identify the summarization and DIAS errors. For example, the shift of SF-1 and SF-4 to the right is due to (i) the skipped values as can be seen in Figure 6a, 6c and 6e and (ii) the lag of the self-correcting computations in DIAS. The former is clearer for SF-12 in which the steps created by the large send factor are observable in the overall error. Moreover, SF-1 and SF-4 have similar errors as during the DIAS aggregation summarization errors cancel out.

Figure 8 provides an overview of all errors averaged for each send factor and aggregation function. This figure shows some key observations: (i) the DIAS error decreases 44.46%, 55.93% and 32.55% for SUMMATION, AVERAGE and MAXIMUM as the send factor increases from SF-1 to SF-16. (ii) Each step increase in the send factor increases dramatically the summarization error that finally overpasses the DIAS error. (iii) The overall error remains on average the sum of its parts. A distinguishable increase is observed in the high values of send factors for the SUMMATION and AVERAGE.

Figure 9 illustrates the total communication cost for each send factor. Intuitively, if a fewer number of changes in the selected states are reported, then a lower communication cost is imposed as a lower number of outdated aggregators appear during runtime. The difference is critical with the communication cost dropping 15.01% and 49.5% for SF-4 and SF-12 compared to SF-1.

C. Regulating the error trade-offs

The proposed decentralized data analytics system for Internet of Things involves different actors that may have differ-

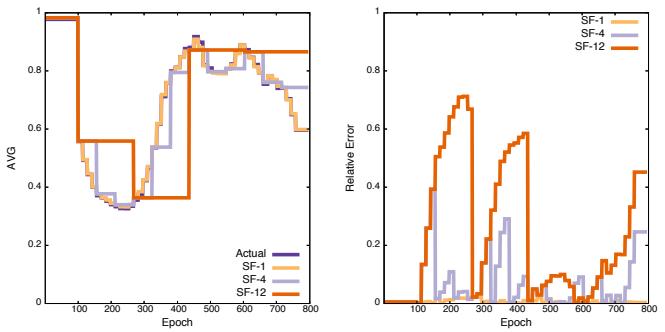


(a) Phone 1, $k = 3$ (b) Phone 1, $k = 5$

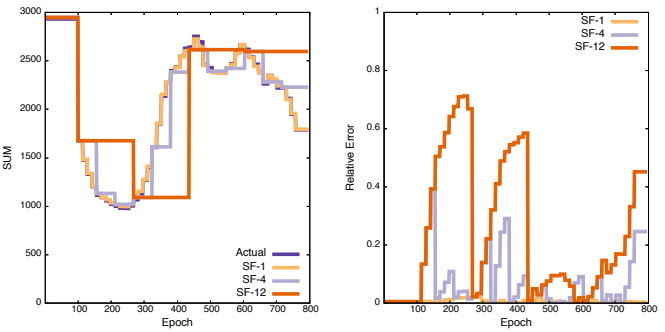
(c) Phone 2, $k = 3$ (d) Phone 2, $k = 5$

(e) Phone 3, $k = 3$ (f) Phone 3, $k = 5$

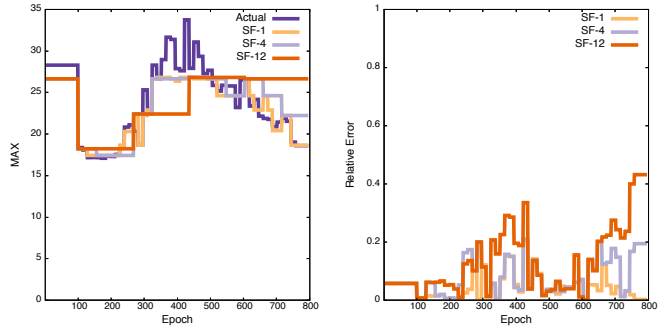
Figure 5. Performance overhead for the three different phones and two different numbers of possible states.



(a) AVERAGE, summarized data (b) AVERAGE, summarization errors



(c) SUMMATION, summarized data (d) SUMMATION, summarization errors



(e) MAXIMUM, summarized data (f) MAXIMUM, summarization errors

Figure 6. Summarized data and summarizaton errors for different send factors and aggregaton functions.

ent requirements for each error involved in the aggregation process. For example, the data suppliers such as users of mobile phones may acquire maximal summarization errors for privacy-preservation and minimal aggregation errors for high quality of service. In contrast, the administrators or system operators of the infrastructure in which DIAS is deployed may require to keep a low bandwidth utilization and therefore choose for the high errors introduced by high send factors. This section illustrates an automated method for decision-making support that encounters the different trade-offs between the three involved errors.

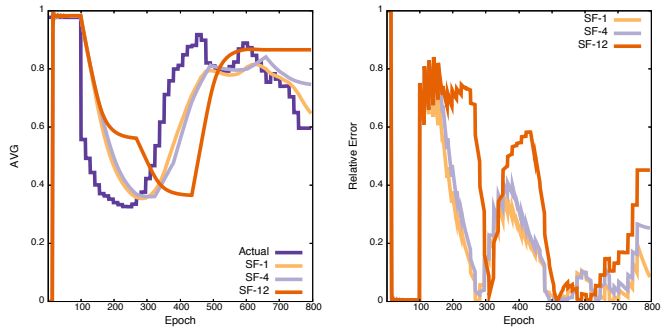
Decision trees built by a Weka implementation of the C4.5 algorithm [24] are used for the decision-making support. Learning is performed by a 10-fold cross-validation of a training set generated with different threshold combinations of an α parameter under a step-wise increment of 0.001. The weight $\alpha \in [0, 1]$ parameter is used as a tolerance preference between two pairs of errors: (i) summarization vs. DIAS errors and (ii) overall vs. DIAS errors. The selected send factor is the one that minimizes the following equation in each case:

$$\epsilon_{s,d} = a\epsilon_s + (a-1)\epsilon_d, \tag{5}$$

$$\epsilon_{o,d} = a\epsilon_o + (a-1)\epsilon_d. \tag{6}$$

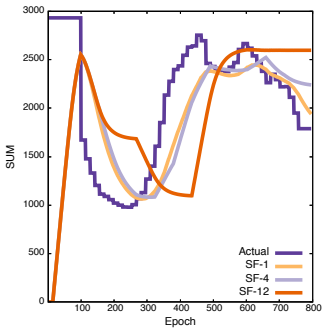
Figure 10 illustrates the decision trees computed.

The decision trees for AVERAGE and SUMMATION are similar in each pair of errors given that they result in similar error values as shown in Figure 8a, 8c, and 8b, 8d. They involve the send factors SF-1, SF-2, SF-4 and SF-16 and the first branching is performed for $\alpha \leq 0.36$ and $\alpha \leq 0.4$ for each pair of errors respectively. In contrast the send factors SF-1, SF-12 and SF-16 appear in the trees for MAXIMUM with the respective first branching performed for $\alpha \leq 0.45$ and $\alpha \leq 0.64$ as shown in Figure 8e, 8f.

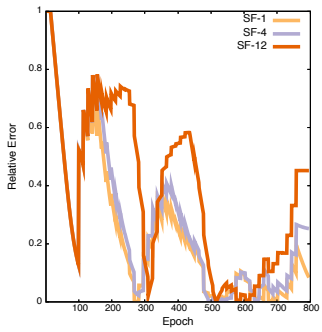


(a) AVERAGE, aggregation data

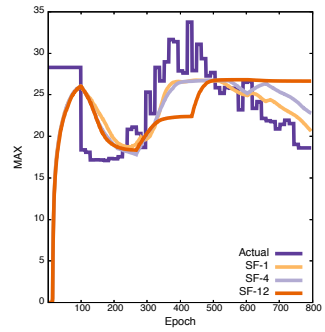
(b) AVERAGE, overall errors



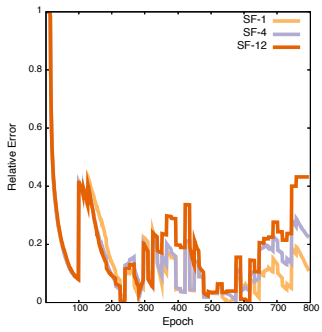
(c) SUMMATION, aggregation data



(d) SUMMATION, overall errors



(e) MAXIMUM, aggregation data



(f) MAXIMUM, overall errors

Figure 7. Aggregation data and overall errors for different send factors and aggregaton functions.

IV. Comparison with Related Work

Data analytics systems can operate in commercial and centrally managed computational resources or crowdsourced distributed resources collectively managed by citizens. Moreover, data processing can be centralized using large and expensive computational resources such as supercomputers and GPUs or it can be performed in a distributed fashion among remote and interactive computational resources. This section focuses on systems and mechanisms, whose system design is fully decentralized for both of the aforementioned aspects. This focus underlines some significant work done for in-network decentralized aggregation using gossiping protocols or tree topologies and some more advanced gossip-based learning.

Gossip-based aggregation [25] provides fast computations of certain aggregation functions in large-scale decentralized networks. Gossiping requires a constant communication cost. Given that convergence of the estimated aggregates to the actual values occurs in a few algorithm repetitions, the communication cost can be decreased with a stopping criterion. However, it is a challenge to come up with a general stopping criterion that works effectively for different network settings and input data. Moreover, gossip-based aggregation relies on the principle of variance reduction in each executed step that limits the aggregation functions to the AVERAGE. Other functions such as the MAXIMUM, MINIMUM, COUNT and SUMMATION can be computed with variations such as applying in the inverse birthday paradox and running multiple instances of the protocol, in contrast to the proposed approach in which a broad range of aggregation functions can be computed within the main system operation. Moreover, gossip-based aggregation protocols are not designed to work with dynamic input values and therefore recomputations need to be performed when any input data changes resulting in higher communication costs and additional complexity for scheduling recomputations. Because of these limitations, gossip-based aggregation is often contrasted with aggregation over tree topologies [26], [27] that can be more efficient, yet, trees require building and maintenance [28] that can be costly in distributed environments.

There are more recent efforts as well to overcome these limitations of decentralized aggregation methods, however, their applicability to Internet of Things remains a grand challenge. TOP-K is studied under dynamic datasets [29] and group-based aggregation is considered to improve efficiency [30]. Synopsis diffusion can perform duplicate-free data aggregation in wireless sensor networks, however, COUNT and SUMMATION are the main options for aggregation [31]. The dimension reduction methods for collaborative gossip learning [8] draw parallels with the concept of summarization in Internet of Things devices illustrated in this paper. However, this paper additionally studies the dimension reduction of the data updates by introducing the concept of the send factor that regulates the frequency of data changes and therefore, the performance trade-offs of the aggregation. This contribution can open up new ways for more efficient gossip learning on fully distributed Internet of Things data [7].

V. Conclusion and Future Work

This paper concludes that fully decentralized data analytics using highly dynamic real-time sensor data from Internet of Things devices is feasible. Trade-offs between privacy, accuracy and communication cost in aggregation can be regulated in an automated fashion using decision trees. This can bridge the gap of different conflicting requirements of actors involved in decentralized data analytics such as users protecting their privacy, vs. infrastructure operators that regulate computational resources. Experimental evidence using a system implementation on smart phones and real-world Smart Grid data shows that the democratization of Internet of Things data analytics as a public good via the engineering of their decentralization is promising and viable. This alternative approach for data analytics is envisioned for sustainable participatory digital societies built on system trust and privacy-preservation.

Future work includes the evaluation of the proposed system under network changes and failures [32]. The robustness of DIAS in such scenarios is currently work in progress as well as the introduction of differential privacy [33] and homomorphic encryption [34] in the aggregation sessions of DIAS to hide the

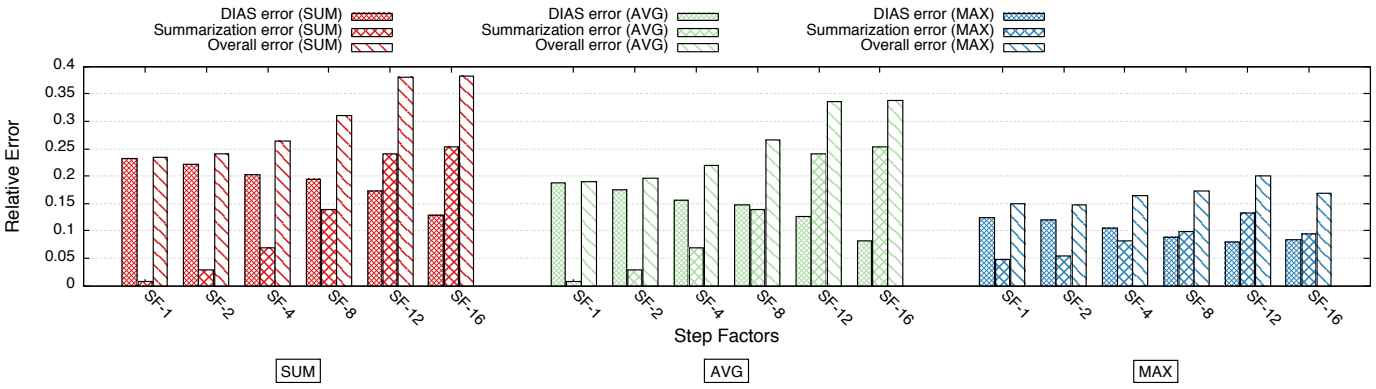


Figure 8. Average relative errors for different send factors and aggregation functions.

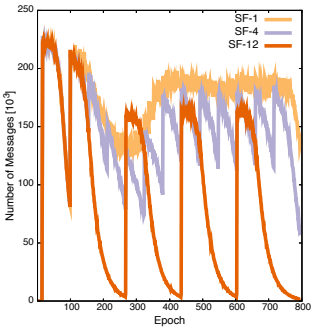


Figure 9. Number of messages sent for different send factors.

exchange of the selected states. Finally, knowledge from game theory on public good games can be utilized to incentivize the formation of a 'network of the commons' for decentralized data analytics as a public good [35], [36], [36].

ACKNOWLEDGMENT

This work is supported by the European Communitys H2020 Program under the scheme INFRAIA-1-2014-2015: Research Infrastructures, grant agreement #654024 SoBigData: Social Mining & Big Data Ecosystem (http://www.sobigdata.eu) and the European Communitys H2020 Program under the scheme ICT-10-2015 RIA, grant agreement #688364 ASSET: Instant Gratification for Collective Awareness and Sustainable Consumerism (http://www.asset-consumerism.eu). The authors would like to thank Pablo Velásquez for the development of the visualization tool with which the network visualizations of Figure 3 are made and Farzam Fanitabasi for his constructive input on the paper.

REFERENCES

[1] K. Thilakarathna, H. Petander, J. Mestre, and A. Seneviratne, "Mobitribe: cost efficient distributed user generated content sharing on smartphones," *IEEE Transactions on Mobile Computing*, vol. 13, no. 9, pp. 2058–2070, 2014.

[2] S. Hajian, F. Bonchi, and C. Castillo, "Algorithmic bias: From discrimination discovery to fairness-aware data mining," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 2125–2126.

[3] D. Helbing and E. Pournaras, "Society: Build digital democracy," *Nature*, vol. 527, pp. 33–34, 2015.

[4] H. Herodotou, H. Lim, G. Luo, N. Borisov, L. Dong, F. B. Cetin, and S. Babu, "Starfish: A self-tuning system for big data analytics." in *CIDR*, vol. 11, 2011, pp. 261–272.

[5] J. Dittrich and J.-A. Quiané-Ruiz, "Efficient big data processing in hadoop mapreduce," *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 2014–2015, 2012.

[6] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, "Trends in big data analytics," *Journal of Parallel and Distributed Computing*, vol. 74, no. 7, pp. 2561–2573, 2014.

[7] R. Ormándi, I. Hegedűs, and M. Jelasity, "Gossip learning with linear models on fully distributed data," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 556–571, 2013.

[8] A. Berta, I. Hegedus, and M. Jelasity, "Dimension reduction methods for collaborative mobile gossip learning," in *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, Feb 2016, pp. 393–397.

[9] E. Pournaras, M. Vasirani, R. E. Kooij, and K. Aberer, "Decentralized planning of energy demand for the management of robustness and discomfort," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2280–2289, 2014.

[10] J. He, D. J. Miller, and G. Kesidis, "Latent interest-group discovery and management by peer-to-peer online social networks," in *Social Computing (SocialCom), 2013 International Conference on*. IEEE, 2013, pp. 162–167.

[11] S. Sevignani, "The problem of privacy in capitalism and the alternative social networking site diaspora," *tripleC: Communication, Capitalism & Critique. Open Access Journal for a Global Sustainable Information Society*, vol. 10, no. 2, pp. 600–617, 2012.

[12] P. Antoniadis and I. Apostol, "The right (s) to the hybrid city and the role of diy networking," *The Journal of Community Informatics*, vol. 10, no. 3, 2014.

[13] A. Sathiaseelan, L. Wang, A. Aucinas, G. Tyson, and J. Crowcroft, "Scandex: Service centric networking for challenged decentralised networks," in *Proceedings of the 2015 Workshop on Do-it-yourself Networking: An Interdisciplinary Approach*, ser. DIYNetworking '15. New York, NY, USA: ACM, 2015, pp. 15–20.

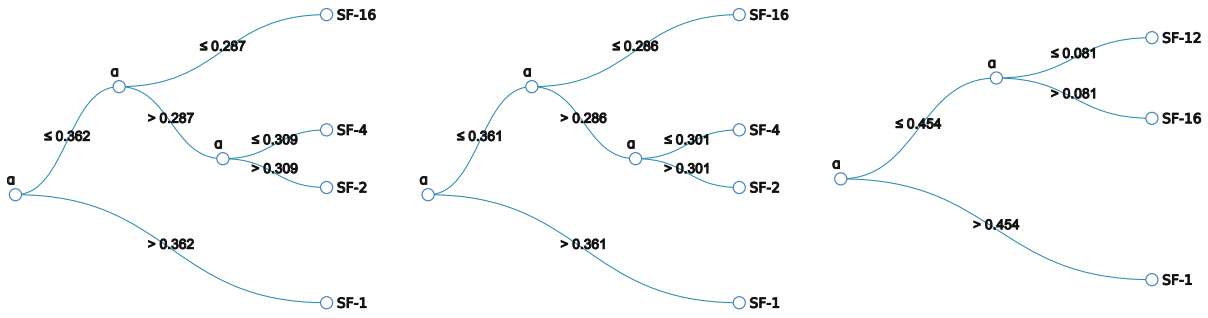
[14] M. Bor, J. E. Vidler, and U. Roedig, "Lora for the internet of things," in *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*. Canada: Junction Publishing, 2016, pp. 361–366.

[15] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "The bittorrent p2p file-sharing system: Measurements and analysis," in *International Workshop on Peer-to-Peer Systems*. Springer, 2005, pp. 205–216.

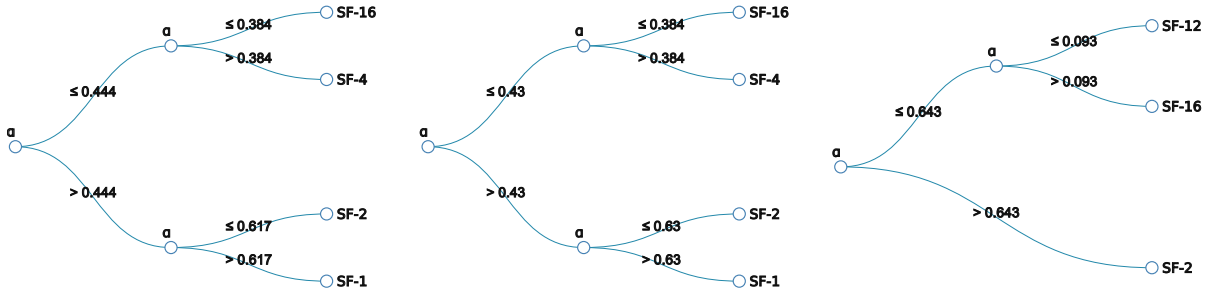
[16] E. Pournaras, M. Warnier, and F. M. Brazier, "A generic and adaptive aggregation service for large-scale decentralized networks," *Complex Adaptive Systems Modeling*, vol. 1, no. 1, p. 1, 2013.

[17] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. Van Steen, "Gossip-based peer sampling," *ACM Transactions on Computer Systems (TOCS)*, vol. 25, no. 3, p. 8, 2007.

[18] E. Pournaras, J. Nikolic, P. Velásquez, M. Trovati, N. Bessis, and



(a) AVERAGE, summarization vs. DIAS error (b) SUMMATION, summarization vs. DIAS error (c) MAXIMUM, summarization vs. DIAS error



(d) AVERAGE, overall vs. DIAS error (e) SUMMATION, overall vs. DIAS error (f) MAXIMUM, overall vs. DIAS error

Figure 10. Decision trees for each aggregation function. The parameter $\alpha \in [0, 1]$ represents the tolerance preference between the two errors.

D. Helbing, "Self-regulatory information sharing in participatory social sensing," *EPJ Data Science*, vol. 5, no. 1, p. 1, 2016.

[19] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping multidimensional data*. Springer, 2006, pp. 25–71.

[20] C. V. C. Bouten, K. T. M. Koekkoek, M. Verduin, R. Kodde, and J. D. Janssen, "A triaxial accelerometer and portable data processing unit for the assessment of daily physical activity," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 3, pp. 136–147, March 1997.

[21] E. Pournaras, I. Moise, and D. Helbing, "Privacy-preserving ubiquitous social mining via modular and compositional virtual sensors," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*. IEEE, 2015, pp. 332–338.

[22] S. Bosse, "Distributed machine learning with self-organizing mobile agents for earthquake monitoring," in *The 2nd International Workshop on Data-driven Self-regulating Systems*. Augsburg, Germany: IEEE, September 2016.

[23] W. Galuba, K. Aberer, Z. Despotovic, and W. Kellerer, "Protopeer: a p2p toolkit bridging the gap between simulation and live deployement," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009, p. 60.

[24] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[25] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *ACM Transactions on Computer Systems (TOCS)*, vol. 23, no. 3, pp. 219–252, 2005.

[26] F. Wuhib, M. Dam, R. Stadler, and A. Clem, "Robust monitoring of network-wide aggregates through gossiping," *IEEE Transactions on Network and Service Management*, vol. 6, no. 2, pp. 95–109, June 2009.

[27] L. Nyers and M. Jelasity, "A comparative study of spanning tree and gossip protocols for aggregation," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 16, pp. 4091–4106, 2015.

[28] E. Pournaras, M. Warnier, and F. M. Brazier, "Adaptive self-organization in distributed tree topologies," *International Journal of Distributed Systems and Technologies (IJDST)*, vol. 5, no. 3, pp. 24–57, 2014.

[29] A. Guerrieri, A. Montresor, and Y. Velegrakis, "Top-k item identification on dynamic and distributed datasets," in *European Conference on Parallel Processing*. Springer, 2014, pp. 270–281.

[30] S. G. Falavarjani, B. T. Ladani, and S. Ghasemi, "Ggra: A grouped gossip based reputation aggregation algorithm," *The ISC International Journal of Information Security*, vol. 7, no. 1, pp. 59–74, 2015.

[31] S. Roy, M. Conti, S. Setia, and S. Jajodia, "Secure data aggregation in wireless sensor networks," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 1040–1052, 2012.

[32] P. Jesus, C. Baquero, and P. S. Almeida, "Flow updating: Fault-tolerant aggregation for dynamic networks," *Journal of Parallel and Distributed Computing*, vol. 78, pp. 53–64, 2015.

[33] S. Niro, J. Lpez, D. Westhoff, and A. Christ, "A keyless gossip algorithm providing light-weight data privacy for prosumer markets," in *Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2015 IEEE International Conference on*, Sept 2015, pp. 31–36.

[34] M. Alaggan, S. Gambs, and A.-M. Kermarrec, *Private Similarity Computation in Distributed Systems: From Cryptography to Differential Privacy*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 357–377.

[35] J. A. Fairfield and C. Engel, "Privacy as a public good," *Duke LJ*, vol. 65, p. 385, 2015.

[36] M. Chessa, J. Grossklags, and P. Loiseau, "A game-theoretic study on non-monetary incentives in data analytics projects with privacy implications," in *2015 IEEE 28th Computer Security Foundations Symposium*. IEEE, 2015, pp. 90–104.