Scalable Cyber-Security Analytics with a New Summary-based Approximate Query Engine

Dominik Ślęzak*, Agnieszka Chądzyńska-Krasowska†,
Joel Holland‡, Piotr Synak‡, Rick Glick§ and Marcin Perkowski‡
*Institute of Informatics, University of Warsaw
ul. Banacha 2, 02-097 Warsaw, Poland
Email: slezak@mimuw.edu.pl
†Polish-Japanese Academy of Information Technology
ul. Koszykowa 86, 02-008 Warsaw, Poland
Email: honzik@pjwstk.edu.pl
‡Security On-Demand
12121 Scripps Summit Dr 320, San Diego, CA 92131, USA
Email: jholland@securityondemand.com
ul. Krzywickiego 34 pok. 215, 02-078 Warsaw, Poland
Email: {marcin.perkowski,piotr.synak}@securityondemand.com
§Independent Consultant
Email: rick.glick@gmail.com

***Abstract*—A growing need for scalable solutions for both machine learning and interactive analytics exists in the area of cyber-security. Machine learning aims at segmentation and classification of log events, which leads towards optimization of the threat monitoring processes. The tools for interactive analytics are required to resolve the uncertain cases, whereby machine learning algorithms are not able to provide a convincing outcome and human expertise is necessary.**

In this paper we focus on a case study of a security operations platform, whereby typical layers of information processing are integrated with a new database engine dedicated to approximate analytics. The engine makes it possible for the security experts to query massive log event data sets in a standard relational style. The query outputs are received orders of magnitude faster than any of the existing database solutions running with comparable resources and, in addition, they are sufficiently accurate to make the right decisions about suspicious corner cases.

The engine internals are driven by the principles of information granulation and summary-based processing. They also refer to the ideas of data quantization, approximate computing, rough sets and probability propagation. In the paper we study how the engine's parameters can influence its performance within the considered environment. In addition to the results of experiments conducted on large data sets, we also discuss some of our high level design decisions including the choice of an approximate query result accuracy measure that should reflect the specifics of the considered threat monitoring operations.

***Index Terms*—Network Event Log Analytics, Approximate Query Engines, Summary Data Processing, Information Granulation, Fuzzy Similarity of Complex Objects**

I. Introduction

A growing need to explore big data sets exists, particularly, in applications involving the machine-/service-generated data. Most companies address this challenge by scaling out computational power even though this strategy is increasingly inefficient. At the same time, people begin to realize that the analytical tasks could be performed in at least partially approximate fashion. This thought process opens new opportunities to search for a balance between the speed, the resource consumption and the accuracy of computations.

The above trend can be observed even in the fields that might be regarded as the most "approximation-prohibitive". Let us refer to the area of cyber-security discussed in this paper. Although one could expect that the network security monitoring operations should rely on detailed analytical processes, the speed of decision-making is sometimes more important than perfect calculations. More precisely, thorough examination is still needed to verify the characteristics of each potential threat. However, the threats need to be first efficiently identified to narrow down further exploration.

Identification of threats can be conducted in many ways. A number of statistical learning and machine intelligence methods can assist in heuristic classification of threats based on patterns and examples learned from the data. On the other hand, there are security specialists who are able to go beyond the observed data and make thoughtful decisions based on a wider context. These two ways need to be combined to build a modern security operations platform. Human experts who use the platform to provide security services need to have truly efficient access to information comprising both the results of machine learning methods and the atomic data.

In this paper we discuss the online *SuperScale Analytics* platform introduced by Security On-Demand to provide such efficient access for both analysts and end-users. The platform is comprised of several data collection, indexation and aggregation layers, with an extra capability of fast ad-hoc interaction with raw network event logs stored in a relational form. In-

978-1-5386-2715-0/17/$31.00 ©2017 IEEE

formation about event logs is actually available in two modes: 1) using standard queries that are advisable for operations on relatively small sets of rows and 2) approximate queries that can perform fast enough for arbitrary data portions.

Approximate query engines can be developed in several ways to assist the organizations specialized in providing their clients with the tools for this kind of online data analytics. Given the requirements of this particular platform, we deployed an engine that conducts approximate data operations based on granulated summaries of the input data. The engine comprises the layers responsible for software agent style acquisition of data summaries and utilizing the stored summaries to produce fast approximate answers to ad-hoc queries. Namely, for each `SQL SELECT` statement, subsequent operations scheduled within its execution plan are performed as transformations of granulated summaries representing their inputs into the summaries representing their outputs.

Embedding an approximate query engine into a production-ready application environment required a lot of tuning and testing. The first task was to design a query accuracy measure that – in the particular area of network monitoring – would perceptually correspond to the user needs and expectations. The applied measure should expose especially (and exclusively) those differences between exact and approximate results of the `SELECT` statements that might mislead the decision makers in their threat assessments. The choice of an appropriate measure was not easy as it needed a compromise between different user groups and different query outcome scenarios. On the other hand, a well-defined accuracy measure was truly needed to find the right trade-off between the performance and exactness of query-driven analytics at the considered platform.

As the next step, we investigated the correspondence between the accuracy observed for some typical ad-hoc queries and the level of granulation of the acquired data summaries. In the considered engine the incoming data are partitioned into collections of rows described by means of single-column frequencies and some multi-column co-occurrence regularities. Thus, operating with larger clusters leads to acceleration of computations but it may also yield less precise data summaries and, consequently, less accurate approximate query results. On the other hand, one could make the summaries more thorough e.g. by increasing the amounts of explicitly represented values and co-occurrence coefficients. However, this may also lead toward higher computational cost of the summary transformation operations that the considered engine relies on.

The rest of the paper is structured as follows. Section II refers to some related works in the considered areas. Section III introduces the SuperScale Analytics platform with a special emphasis on the embedded summary-based query engine. Section IV presents our research related to approximate query accuracy measures and justifies the similarity-based formula that we finally decided with which to proceed. Section V reports some of our experiments conducted on massive real life network event data sets with respect to searching for the right granulation settings for the considered data summaries. Section VI concludes our work with final remarks.

II. Related Work

One can see a growing importance of statistical modeling and machine learning methods in the field of cyber-security, with well-established specific areas such as adaptive anomaly detection and intrusion type classification [13], [27]. In particular, there are many approaches to analyzing the network log event data sets that are based on the principles of Bayesian modeling and reasoning, in relation to Bayesian-style graphical models and estimation methods [4], [16]. Moreover, new computer security solutions begin to adapt some ideas taken from computational intelligence including, e.g., the elements of granular computing and rough sets [20], [21].

There are also some works focused on providing the users of cyber-security systems with visually-supported tools for data interaction [10], [24]. Moreover, it is possible to actively search through meaningful information sources and integrate the acquired data with network-security-related knowledge bases [3], [29]. This way the developers of security analytics platforms can take the advantage of both knowledge discovery techniques that enrich the original data with insightful information and visual exploration techniques that let domain experts efficiently investigate the toughest cases.

From the perspective of the SuperScale Analytics platform, it is also worth referring to the methods supporting incremental data exploration, whereby the users begin their interaction with the system at the level of general trends and then they drill down to examine more detailed aspects of available information. Such "zooming-in" processes are often empowered by analytical database solutions that resolve queries triggered by visual interfaces. Some of those solutions assume that query results obtained by the users at the beginning of exploration do not need to be fully exact or – in some other scenarios – they do not need to be exact at once [12], [14].

This way we enter a rapidly emerging market of solutions aimed at approximate data analytics and particularly, approximate database engines [6], [19]. Although there is still a lot to be done to set up the right expectations and fully integrate such engines with the mainstream of information technology applications, including the field of cyber-security, it is tempting to provide the users with faster query answers even at the cost of loosening their accuracy. It is especially important in the scenarios where the speed of reaction really matters and where slight inexactness of analytical query results is not likely to damage the quality of final decision-making.

The query layer that empowers the SuperScale Analytics platform works with intelligently produced data summaries. The platform connects with two engines – approximate and standard [25], [26]. The summary-based approaches to data processing are already well-known in the literature [1], [9]. However, both considered engines rely on summaries that are built in a unique way, basing on the ideas of data quantization and granulation. Herein, we refer to the paradigms of approximate computing and already-mentioned granular computing [2], [30]. We believe that these two paradigms will play an increasing role in the area of big data analytics.

TABLE I
DATA TABLE WITH ATOMIC NETWORK EVENT LOGS.

Column name: ***Column description***
srcipint: *integer version of IP V4 address of the source*
srcport: *source port used for communication by the operating system*
dstipint: *integer version of IP V4 address of the destination*
dstport: *destination port used for communication by the application*
reportdevice: *device that is logging the event*
devicetypename: *vendor device type*
devicevendor: *device vendor that produced the appliance*
subjectuser: *user account conducting an action*
clientid: *client identification*
srccountrycd: *country code for the source of communication*
dstcountrycd: *country code for the destination of communication*
reportdeviceint: *integer version of IP V4 address of the reporting device*
targetuser: *user account that an action is performed against*
timestampday: *integer version of YYYYMMDD timestamp*
timestamphour: *integer version of YYYYMMDDHH timestamp*
timestampminute: *integer version of YYYYMMDDHHMM timestamp*
timestampsecond: *integer version of YYYYMMDDHHMMSS timestamp*
direction: *direction of an event*
disposition: *whether an event was allowed, denied or unknown*
signatureid: *ID representation of a known signature*
assetid: *ID representation of a known asset*
userid: *ID representation of a known user*
networkid: *ID representation of a known network segment*
devid: *device specific event identifier*
eventname: *device specific event name*
protocol: *protocol used for network communication*
asnid: *identifier for the Autonomous System Number*

III. SUPERSCALE ANALYTICS PLATFORM

As data sets are getting larger and hackers increasingly sophisticated, adding more and more computational power to identify breaches is no longer scalable. This is especially visible in situations when modern machine intelligence techniques are not able to assist the users. One of the current trends in decision-making is actually to configure the utilized classification and prediction models in such a way that they produce final scores only for sufficiently certain cases, leaving the "boundary" for further investigation by humans [21], [30]. This is especially worth considering in the application areas whereby the cost of false-positive/negative mistakes is truly high, like in the world of cyber-security [4], [27].

The SuperScale Analytics platform delivered by Security On-Demand (SoD) follows the above approach. SoD provides a wide range of tools that annotate and aggregate massive amounts of the event log data. These tools make it possible for the security experts to work with the alert tables, where the rows correspond to sequences of network connections and the columns correspond to different types of potential threat indicators [16], [29]. Reliable decisions can be made quite often at such an aggregated level. However, sometimes it is necessary to interrogate the original atomic data. The platform discussed in this paper addresses this particular need.

Table I describes columns in the relational data table gathering network events that SoD collects for each of its customers. Given the intensity of monitored traffic, the data growth is observed at the level of over 300 billions of new rows per month. To provide its security analysts with fast data access, SoD relied on the database solution reported in [26], now available as *InfobrightDB*. Recently, it was decided to provide similar access to much wider group of end-users, so the customers can conduct ad-hoc analytics also by themselves. The expectation for an average query response time was set at the level of two seconds. As such expectation is impossible to meet by any kind of standard database software using reasonable resources, SoD decided to leverage the summary-based approximate query engine introduced in [25].

Figure 1 illustrates the online user interface. Any additional condition or a split with respect to one of data columns yields the corresponding `SQL` statement. The platform is configured to connect with the standard database engine to run queries that are expected to be highly selective. Such queries are generated in the end of the "zooming-in" process, when the user wants to derive complete information about specific cases observed in the data. The approximate database engine is employed for queries that need to scan through larger data portions, as it would take too long to wait for their exact results. These are rather exploratory queries triggered at the stage of "looking around", so their outputs do not need to be perfectly precise. Nevertheless, some reasonable accuracy criteria need to be met. In particular, the approximate engine should not produce too many *false absence* outcomes, i.e., the zeroed `COUNT(*)` results that would be highly positive in "the reality".

The discussed engine captures information in a form of single- and two-column summaries. It composes groups of the newly loaded data rows and constructs summaries for each group separately. Its query execution mechanisms do not assume any access to the original groups. Those groups can be available in a broader framework – like it happens for SuperScale Analytics – but the goal of this particular engine is to work with summaries. For a query received from the platform, each consecutive data operation (such as filtering, grouping, etc.) scheduled in the query execution plan is performed as transformation of summaries representing its input into summaries representing its output. Thus, our summaries can be interpreted as information granules, while the process of their transformation can be treated as an example of industry realization of the paradigms of granular computing.

The designers of the SuperScale Analytics platform investigated also some other approximate query solutions. While the data-sampling techniques turned out to be problematic because of their limited capability to scale and a tendency to produce too many of the above-mentioned *false absence* results, the summary-based approaches seemed to be more promising [9]. Given the query speed requirements outlined in Section I, it was crucial to perform all heavy-duty calculations *entirely* at the level of summaries. We achieved it by harnessing several methods known from other fields, e.g., the mechanism of tree-based belief propagation to populate the `WHERE`-related changes in summaries [22] and the principles of rough set approximations to compose granulated outcomes of `GROUP BY` [8]. Moreover, to support the trade-off between performance and accuracy of query results, we developed a parameterized framework for quantized data summarization [7]. We refer to those parameters in our experiments in Section V.

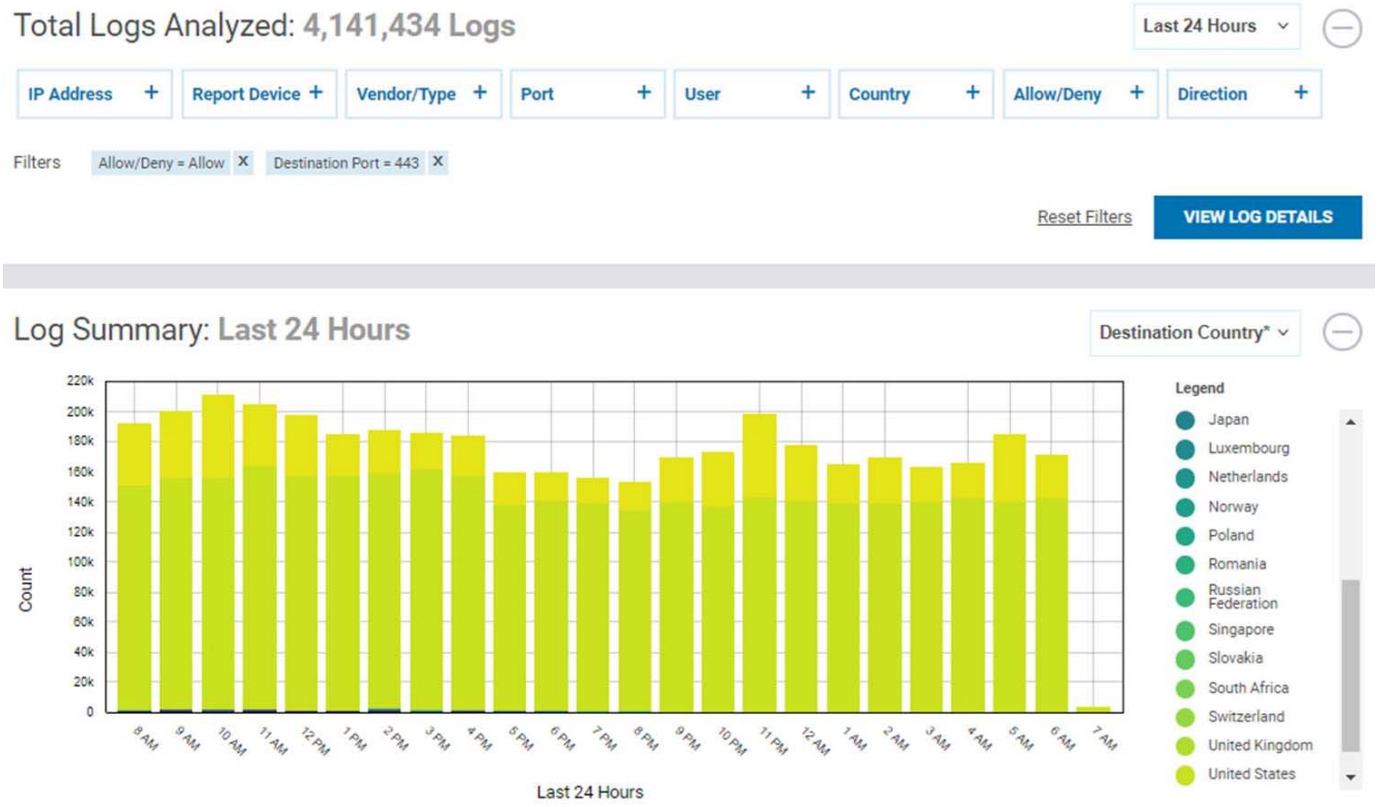

Fig. 1. A screenshot from the SuperScale Analytics online interface. The number of rows in the queried table that satisfy conditions `disposition = Allow` and `dstport = 443` is approximately equal to 4,141,434. The colored bars reflect approximate counts for the groups defined with respect to `dstcountrycd` and `timestamphour` for these conditions.

IV. Similarity-Based Query Accuracy

In this section we outline our approach for expressing the *accuracy* of the `SELECT` statement results produced by an approximate database. We propose to do it by measuring the *similarity* between approximate and exact results, somewhat by analogy to the methods of defining fuzzy-set-based hierarchical similarities between complex objects [28]. The considered mathematical procedure can be applied for evaluation of an arbitrary approximate query engine. However, our primary inspiration relates to the engine introduced in [25] and its application field discussed in Section III.

Figure 2 illustrates a simple example of calculation of the query result similarity. It refers to so-called "top n" queries that group the data with respect to some dimensions and report only a certain number of groups with the highest aggregation scores. More generally, the result of every `SELECT` statement takes a tabular form, herein denoted as $R = (U, A)$, where U and A are the sets of its tuples and attributes. U can refer to original rows, groups, etc., while A can refer to data columns (for `SELECT a,b,c...` without `GROUP BY`) or aggregate functions occurring after `SELECT` (for an aggregate query without the `GROUP BY` clause U contains a single element). For the statements including joins, subqueries, expressions, window functions, etc., the meanings of U and A can be introduced in quite a straightforward way as well.

Let $\widetilde{R} = (\widetilde{U}, A)$ denote the approximate result of a given query. For `GROUP BY`, the elements of U and $\widetilde{U}$ belong to the same domain of vectors of values of the grouping columns. If we wanted to compute the similarity between R and $\widetilde{R}$ just by means of occurrences of groups, the corresponding formula might take a form of the ratio $\frac{|U \cap \widetilde{U}|}{|U \cup \widetilde{U}|}$. However, we would like to assess also the similarity of R's and $\widetilde{R}$'s attribute values over the *matched* groups. Thus, for any $u \in U \cap \widetilde{U}$, we investigate the similarity score $sim(A(u), \widetilde{A(u)})$, where $A(u)$ and $\widetilde{A(u)}$ are the vectors of u's values in R and $\widetilde{R}$, respectively. Such score can be calculated as a t-norm of atomic similarities $sim(a(u), \widetilde{a(u)})$, $a \in A$, considered for the corresponding pairs of attribute values in R and $\widetilde{R}$. In Figure 2, the Zadeh's t-norm is applied. Finally, the overall similarity $Sim(R, \widetilde{R})$ is obtained by replacing $|U \cap \widetilde{U}|$ by the aggregate score $\sum_{u \in U \cap \widetilde{U}} sim(A(u), \widetilde{A(u)})$ in the above ratio.

Surely, this is just one of many ways to calculate the approximate result accuracies. For an alternative approach let us refer to [15], where one can also find some early attempts to run approximate queries by means of histogram transformations – which is one of the key ideas of the engine considered in this paper too. As yet another source of inspiration, let us point at [5] – a thorough overview of similarity measures for probability distributions represented using histograms.

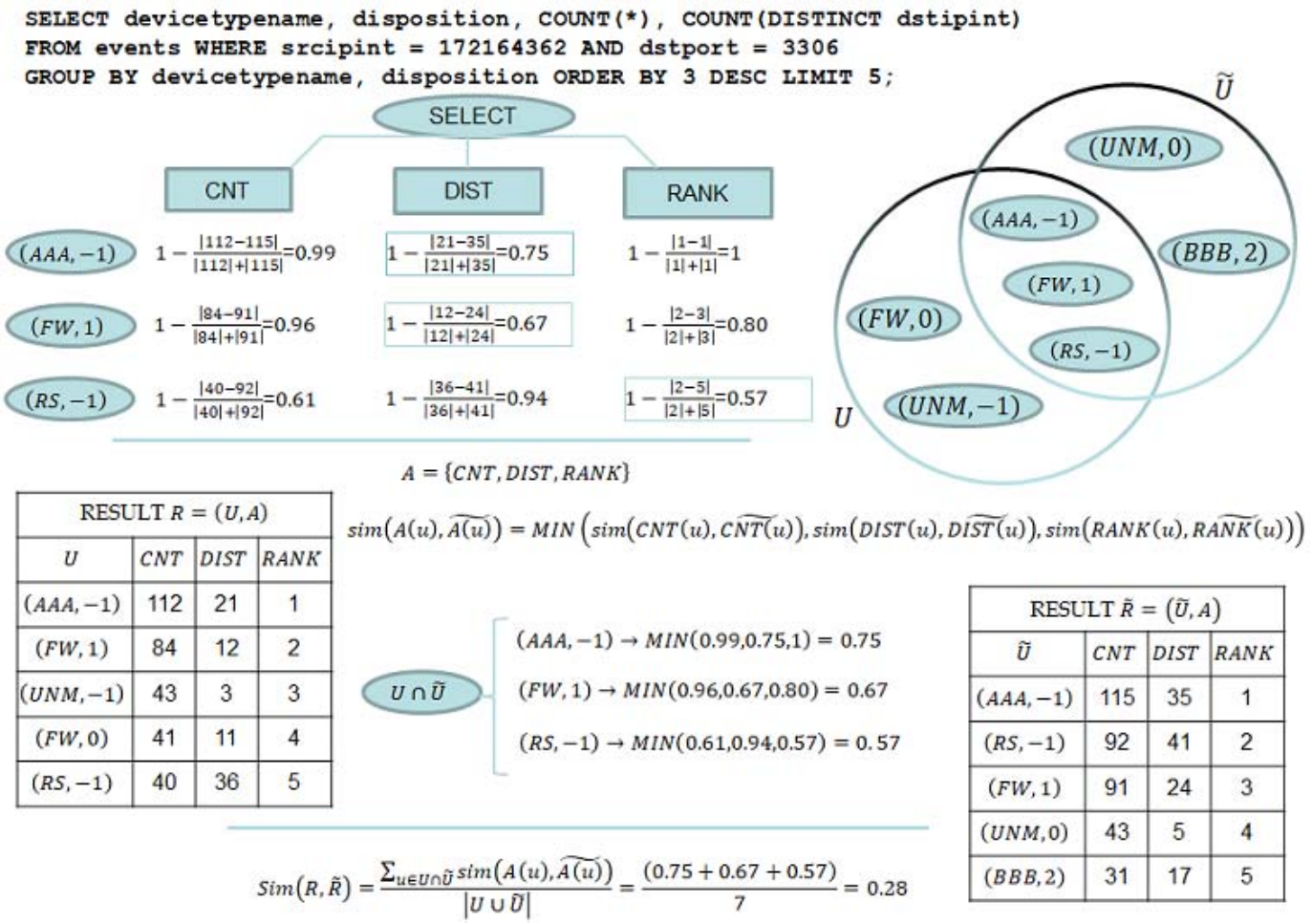

Fig. 2. Comparison of the exact and approximate query results for the case of `GROUP BY` statement. The results are interpreted as information systems with the objects uniquely defined by combinations of values of columns `devicetypename` and `disposition`. The attributes correspond to aggregates `COUNT(*)` and `COUNT(DISTINCT dstipint)`, with the additional attribute `RANK` that reflects the aspect of `ORDER BY`.

Figure 2 displays also the atomic similarity function that we use in our experiments to compare the exact and approximate query outputs at the level of single values (note that we put $sim = 0$ if one of results is `NULL` and the other is not):

$$sim(a(u), \widetilde{a(u)}) = \begin{cases} 1 & \text{for } a(u) = \widetilde{a(u)} \\ 1 - \frac{|a(u) - \widetilde{a(u)}|}{|a(u)| + |\widetilde{a(u)}|} & \text{otherwise} \end{cases} \quad (1)$$

This measure corresponds to a single-dimensional version of the Canberra distance, which was considered in some approaches to intrusion detection [11], as well as in other areas, e.g., to express similarities between partial rankings [17]. In particular, although that latter publication refers to the field of bioinformatics, it is a useful guidance how to evaluate approximate outcomes of the "top n" queries that can be interpreted by means of partial rankings as well.

A task of choosing the appropriate measure at this level is not easy, as it needs to meet the user expectations in the given application area. Such expectations can be expressed as explicit requirements related to similarity properties or implicit intuitions related to a kind of *perceptual closeness* of the exact and approximate results. With this respect we conducted several surveys involving business analysts, network security experts and database practitioners. We did it for the simplest form of analytical ad-hoc queries – the `SELECT COUNT(*) FROM t WHERE conditions` statements.

The first survey concentrated on the preferred mathematical properties. We assumed that all eligible similarity scores $f(\textit{exact count}, \textit{approximate count})$ should satisfy $0 \leq f(x, y) \leq 1$ and $f(x, x) = 1$. We asked the survey participants about the following properties, among others:

1. $f(x, y) = 1 \Leftrightarrow x = y$
2. $f(x, y) = f(y, x)$
3. $\lim_{y \to +\infty} f(x, y) = 0$
4. $x > 0 \Rightarrow f(x, 0) = f(0, x) = 0$

Properties 1 and 2 were evaluated as "neutral". Property 3 was commonly wanted as reflecting the increasing approximation error. Property 4 led to diverse opinions among the database experts. (Some of them could not accept the same evaluation of the cases such as $f(0, 1)$ and $f(0, 100)$.) However, it was considered as valid by the security analysts, given their strong aversion to the *false absence/presence* results.

In the second survey, we wanted to understand how the users of an approximate query engine might perceive the concept of *closeness*. We tested whether the exact and approximate counts x and y are regarded as more similar to each other if they are closer with respect to the difference $|x - y|$ and/or have a higher proportion score $\min\{\frac{x}{y}, \frac{y}{x}\}$. The study was conducted over three groups of participants who were fed with 25 pairs of the exact/approximate hypothetical `COUNT(*)` results. We prepared the sets of pairs of 25 small, 25 medium and 25 large results to verify whether human perception depends on the scale of query outcomes. For different sets, the considered pairs were re-scaled linearly with respect to their differences while keeping their proportion scores unchanged.

Table II illustrates the gathered feedback. The participants in each group were requested to order the given 25 pairs by means of their dissimilarity, according to their own intuitions. The obtained orderings were then tested against the baseline orderings corresponding to differences and disproportions. The reported results seem to indicate that the groups of people perceiving dissimilarities as having something in common with differences and disproportions are roughly equal to each other. Therefore, we should look carefully at the similarity scores that refer to both of those aspects of perception.

The above findings confirmed that the similarity function (1) seems to be worth considering. First, it has some relevant background in the literature. Second, it satisfies properties 1-4. Third, it is related to both differences and proportions, by means of the following equality, for $x > 0$, $y > 0$:

$$1 - \frac{|x - y|}{x + y} = \frac{2}{1 + \frac{1}{\min\{\frac{x}{y}, \frac{y}{x}\}}}$$

We prepared analogous surveys for other similarity functions as well. We also examined other modifications of the procedure visible in Figure 2, e.g., with regards to a choice of t-norm. We could see that the approach based on the Zadeh's t-norm and the formula (1) makes sense from both theoretical and empirical perspectives. In particular, property 4 turned out to be compatible with the formula for $Sim(R, \widetilde{R})$. Namely, for the *false presence* and *false absence* groups represented by the sets $\widetilde{U} \setminus U$ and $U \setminus \widetilde{U}$, respectively, the atomic similarity scores are expected to be equal to 0. Thus, there is no need to include them into the aggregate score defined over $U \cap \widetilde{U}$.

On the other hand, our investigations tend to show that there is no single similarity function that reflects all expectations. For example, let us go back to the SuperScale Analytics framework outlined in Section III. According to its designers, it is indeed crucial to tune the approximate query framework to limit the occurrences of *false presence/absence* results. However, these two categories of mistakes are of different importance for different user groups. – *False absences* are harmful for the SoD security analysts who attempt to detect the emerging patterns that did not occur in the past, while *false presences* are misleading for the end-users who narrow down their search by basing on approximate queries and then switch to the exact mode to analyze more details.

TABLE II
OUTCOMES OF THE "DIFFERENCES VERSUS DISPROPORTIONS" SURVEY.

sizes of query results in a survey	**small**	**medium**	**large**
number of participants in three groups	22	24	20
orderings correlated with differences	8	8	8
orderings correlated with disproportions	9	7	6
orderings insufficiently correlated	5	9	6

V. EXPERIMENTS WITH ENGINE PARAMETERS

There are many aspects that influence the speed and accuracy of calculations in the considered engine. To minimize the summary footprint and accelerate the approximate query execution, our granulation algorithms create quantized histograms that do not provide complete representation of all distinct values occurring in the original data. They rather focus on values that look like most interesting (so-called *special values*) and summarize the remainders of locally observed column domains in a range-based fashion. Similarly, we do not store full information about the ratios of *co-occurrence* of values (and ranges) on different columns. Instead, we register a limited number of ratios that seem to be most meaningful. For instance, if two frequent values – v on column `a` and w on column `b` – co-occur relatively rarely (or do not co-occur at all) in a given collection of rows, then this information may be added to the collection's granulated summary.

This kind of imperfect representation required us to redesign all data operations involved in the process of `SQL` execution [25]. We needed also to adjust heuristic methods for choosing the most important special values and co-occurrences that should be stored to increase the expected accuracy of approximate data analytics [7]. In this paper we refer to one more aspect of the engine tuning – a balance between the expected accuracy and the *budgets* specified as the maximum amounts of special values and co-occurrence ratios that can be stored for each collection of the ingested original rows. Moreover, we examine the levels of *granulation resolution*, i.e., the number of rows in each of the summarized data collections.

Before we proceed with our experiments, let us comment on a general task of database benchmarking. A common approach in this area is to use randomly generated data sets and artificial `SQL` statements reflecting typical analytical query workloads [23]. Such frameworks are usually designed to investigate tradeoffs between the applied computational resources and the obtained query performance. However, they can be easily extended towards a new dimension related to the query result accuracy with the assumption that less accurate calculations should be faster, requiring relatively less resources.

A more dedicated approach is to conduct the performance versus accuracy tests on real-world data sets representative for a given application domain. Figure 3 illustrates a scheme of producing a family of simple "diagnostic" analytical queries that can be executed by an approximate database engine to generally assess the accuracy that it can deliver. Herein, we assume the basic knowledge about the input data set with respect to categories of its particular columns [18]:

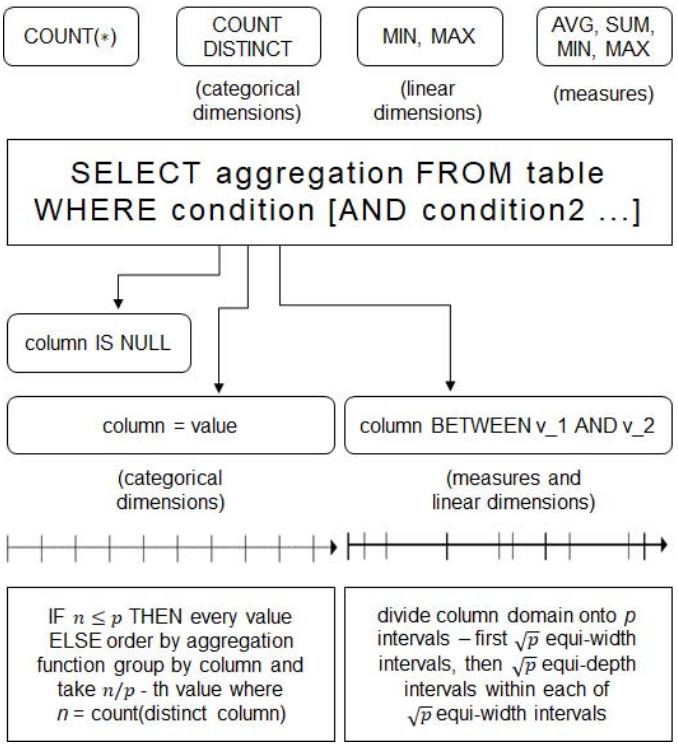

Fig. 3. A general mechanism of producing simple analytical statements that can be used to assess the expected accuracy of an approximate query engine for a given real-world data set in the case of insufficient information about representative examples of real-world queries.

TABLE III
EXAMPLES OF REAL-WORLD QUERIES USED IN THE EXPERIMENTS.

```
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND srcport = 0
AND dstport = 443 AND devicetypename = 'FW';
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND
devicevendor = 'MICROSOFT' AND
dstipint = 1116652493 AND devicetypename = 'AAA';
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND dstipint = 134744072
AND srcport = '443' AND devicetypename = 'FW';
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND
devicevendor = 'MICROSOFT' AND
srcipint = 172164362 AND dstipint = 1116652493;
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND srcipint = 172164362
AND dstipint = 134744072 AND dstport = '3306';
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND
devicevendor = 'Security On-Demand' AND
srcipint = 172164362 AND dstipint = 134744072;
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND srcipint = 171834990
AND dstipint = 134744072 AND srcport = 875;
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND srcipint = 171863829;
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND
srcipint = 171863829 AND dstcountrycd = 'CH';
SELECT COUNT(*) FROM events WHERE
timestampday = 20170410 AND srcipint = 171863829
AND dstcountrycd = 'CH' AND dstport = 80;
```

1) *Measures* that are typically aggregated and/or quantized in the analytical queries
2) *Categorical dimensions* that include foreign keys and their corresponding hierarchies
3) *Linear dimensions* for which it is more natural to operate with the ranges rather than equalities

The most reliable strategy is however to work with both real-world data tables and representative real queries. Therefore, although the considered engine is a fully universal solution for SQL-based approximate analytics [25], it is indeed profitable to focus on its specific use cases in specific application fields, like the one discussed in this paper. Table III includes several examples of the `SELECT COUNT(*)` statements that were triggered by the SoD analysts within the SuperScale Analytics framework. For the testing purposes, we chose 100 of such queries and we measured the accuracy of their approximate outcomes over a relatively small network event data set of the original size of roughly 100 gigabytes. Surely, more experiments with other data sets and analytical query workloads will be still needed. Nevertheless, we believe that the reported results provide some general insights about the expected accuracy of our summary-based approach.

Figure 4 summarizes our experimental findings. The "baseline" refers to the results obtained for the following default settings of the granulation/representation parameters:

1) Every consecutive data collection that needs to be summarized has 2^{16} rows (abbreviated as "64K")
2) For each collection and each column, its quantized representation can include explicit information about up to 100 special values (let us refer to [8] for other parameters of our single-column descriptions, such as the already-mentioned range-based summarizations of the frequencies of "non-special" values and the rough-set-style approximations of column domains)
3) For each collection of rows, the maximum number of co-occurrence ratios that can be stored is equal to 150 times the number of columns in the given table

One can see that the total size of summaries produced with such parameters is roughly 500 megabytes while the average accuracy (1) of approximate results of the above-mentioned 100 queries is slightly higher than 0.61. For comparison, let us take a look at the configuration "256K $\times$ base", which means that the original data set was partitioned into four times larger collections of rows but each of those collections was summarized using the unchanged limits for special values (as well as other components of the single-column representations) and co-occurrence ratios. Such settings yield a lower data summary footprint (as we have four times less collections with the sizes of summaries that are comparable to the previous ones) but also a poorer accuracy (as we attempt to describe larger data portions by using the similar budgets as before).

Table IV outlines all parameter configurations, for which we ran the experiments visible in Figure 4, whereby "256K" and "1024K" denote the number of rows in each single collection

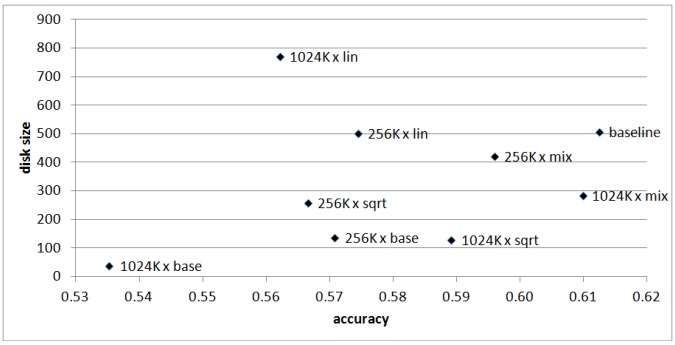

Fig. 4. The total size of data summaries versus the average query accuracy – the results obtained for different data granulation/representation settings.

while "base", "sqrt", "lin" and "mix" correspond to a growth of the allowed amounts of special values and co-occurrence ratios along with the increase in the size of the ingested collections. As before, "base" indicates no budget changes. The meaning of other abbreviations is as follows:

1) "sqrt" – the budgets grow with a square-root-proportion comparing to the sizes of collections
2) "lin" – the budgets grow linearly, so there are practically no savings related to operating with larger collections (unless some columns have truly simple domains that do not require to reach the budget limits)
3) "mix" – the budgets for special values and co-occurrences grow like "sqrt" and "lin", respectively

The above cases are just a few examples of configurations that our summary-based query engine can handle at the production-ready level. Needless to say, the SuperScale Analytics platform is prepared to work with these configurations too.

Footprints displayed in Figure 4 provide us also with some intuitions about the expected speed of approximate query execution. This is because they correspond directly to the lengths of so-called *hot loops* [2] that are performed during the designed approximate computations. Indeed, as explained in [25], the cost of all heavy-duty transformations of the input summaries into the output summaries depends linearly on their level of representation detail (e.g., the number of special values) and, obviously, the number of summary instances (i.e., the number of the considered collections of rows).

From this perspective, one should pay a particular attention to the case of "mix", whereby the footprint/performace savings are expected practically only for the single-column summaries while the computational effort related to operating with the co-occurrence ratios remains the same as for the default settings. This strategy is justified as follows: Consider columns `a` and `b` represented by 200 special values each, for a single collection of 256K rows. Then the number of pairs of special values equals to 200×200 – four times higher than 100×100 in the default engine configuration. Hence, a proportional coverage of the most meaningful co-occurrence ratios seems to require a budget that grows linearly with the square-root-increase of the resolution of the single-column representations.

TABLE IV
MAXIMUM PER-COLUMN×COLLECTION BUDGETS FOR STORING SPECIAL VALUES AND CO-OCCURRENCE RATIOS USED IN THE EXPERIMENTS.

	base	**sqrt**	**mix**	**lin**
256K	special: 100 co-oc: 150	special: 200 co-oc: 300	special: 200 co-oc: 600	special: 400 co-oc: 600
1024K	special: 100 co-oc: 150	special: 400 co-oc: 600	special: 400 co-oc: 2400	special: 1600 co-oc: 2400

TABLE V
DETAILED ACCURACY RESULTS FOR EXPERIMENTS IN FIGURE 4.

		Exact Query Results					
Approximate Results		Abs	Prs	Abs	Prs	Abs	Prs
		256K x sqrt		256K x base		1024K x sqrt	
	Abs	35	3	37	3	38	2
	Prs	32	30 (0.72)	30	30 (0.67)	29	31 (0.67)
		256K x mix		baseline		1024K x mix	
	Abs	38	3	41	5	39	3
	Prs	29	30 (0.72)	26	28 (0.72)	28	30 (0.73)
		256K x lin		1024K x base		1024K x lin	
	Abs	35	3	37	7	34	3
	Prs	32	30 (0.75)	30	26 (0.64)	33	30 (0.74)

Accuracies in Figure 4 are quite low. However, we need to remember that the formula (1) is quite restrictive and, in particular, it puts 0 for any *false absence* or *false presence*. More detailed statistics are reported in Table V. For the "baseline" we observed 41 *true absences* (which means that both approximate and exact results of the corresponding `SELECT COUNT(*)` statements were equal to 0), five *false absences* and 26 *false presences*. Generally, we can see a low number of *false absences*, which is important – as already mentioned – for the security analysts (especially when comparing to the sampling-based approximate query solutions [6]). However, we still need to work on too many *false presences*.

Let us additionally focus on two settings: "1024K × sqrt" and "1024K × mix". Both of them yield 16 times less summaries than the "baseline", although each of summaries has a relatively richer structure. For "1024K × sqrt" the allowed amounts of special values and co-occurrences per summary are four times larger. Still, the overall number of stored information pieces is four times lower than the "baseline", which means four times shorter *hot loops* and lower footprint. On the other hand, the accuracy drops down. It can be seen particularly over the *true presence* cases, whose average accuracies are put into brackets in Table V.

For "1024K × mix" the average accuracy is almost the same as the "baseline", with a significantly lower footprint (although not as low as "sqrt"). It seems to show that information about correlations between occurrences of values on different columns is a bit more important than information about the domains of particular single columns. As mentioned in Section III, our engine executes the multidimensional filtering operations by using an approximate version of the tree-based probabilistic propagation [22]. Hence, richer information about the joint probability distributions (approximated by a limited number of the most meaningful co-occurrence ratios) should indeed yield a higher accuracy of the whole process.

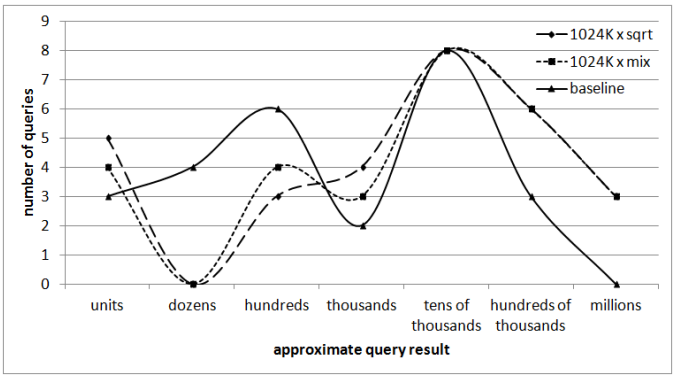

Fig. 5. The approximate counts for the *false presence* cases.

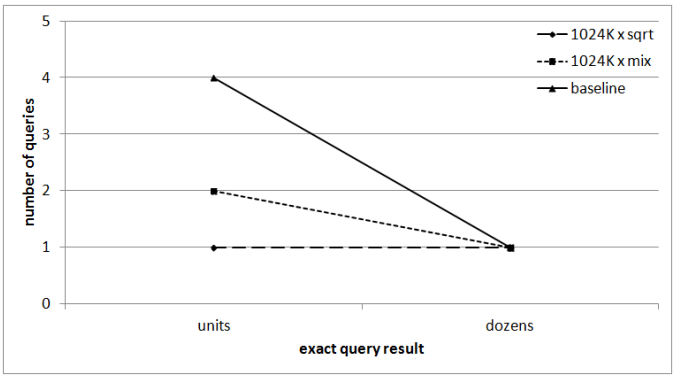

Fig. 6. The exact counts for the *false absence* cases.

The above examples show that the implemented parameters can be indeed used to seek for a balance between the accuracy and the expected speed of resolving analytical queries, approximated using the summary footprint. Actually, the analysis of disk sizes is not the only available tool with this respect. In [8], we introduced a metadata framework that provides the complete information about – among the others – special values and co-occurrence ratios in a convenient relational format. Thus, the specifics of *hot loops* responsible for the query performance can be investigated using `SQL`.

Surely, one should adjust the final tuning of such parameters to a specific application field. As visible in Section IV, the needs of different user groups can be quite diverse. This may trigger further changes in the accuracy formulas or even in the ranking functions utilized at the stage of extracting the most meaningful pieces of information from the original data [7]. Indeed, for a given application framework, the ultimate challenge is to find a "correlation" between the heuristic measures employed to construct data summaries and the user preferences when it comes to query accuracies.

For example, let us go back again to Table V and take a closer look at *false presences* and *false absences*. Although the formula (1) treats all such cases equally, in some practical scenarios it may be worth distinguishing between "large" and "small" mistakes of this kind (see also the discussion about $f(0,1)$ versus $f(0,100)$ in Section IV). Such an additional analysis does not mean that the considered similarity function is wrong. We rather suggest that it may be useful to look at the engine's accuracy from multiple perspectives.

Figures 5 and 6 display the ranges of, respectively, approximate `COUNT(*)` results that should be equal to 0 and exact `COUNT(*)` results that are mistakenly computed as equal to 0, for the settings "1024K $\times$ sqrt" and "1024K $\times$ mix" versus the "baseline". There are generally no issues with *false absences* – the largest exact query result in this category does not exceed the level of 100. However, the other case is more problematic. In particular, for "1024K $\times$ mix" there are three queries with approximate outcomes counted wrongly in millions. Thus, although this configuration seemed to be quite reasonable basing Figure 4, it requires further analysis because of the occurrence of "very large" *false presences*.

VI. Conclusions And Future Directions

We presented the SuperScale Analytics platform – designed for the purposes of exploration of massive sets of network event logs – that is empowered by our `SQL`-based approximate query engine relying on scalable transformations of granulated data summaries. While referring to [25] for more detailed description of the processes of intelligent summary construction and approximate query execution, we concentrated on establishing a parameterized framework for empirical evaluation of the engine's accuracy, footprint and performance. We discussed our way of calculating the accuracy as a similarity between the approximate and exact versions of the `SELECT` statement results. Finally, we conducted some experiments with real data sets and queries in order to connect the granularity/resolution parameters used while creating summaries of the atomic data with the average expected accuracy of approximate queries executed over those summaries.

Surely, there are still some important aspects that we did not discuss in this paper. First, we focused on evaluating the accuracy by means of measuring similarities between the exact and approximate analytical query results. We regard it as the first step towards the empirical analysis of a given approximate engine. However, in the literature there are many works devoted to the confidence of approximate query results [19]. This is a different topic, as so-called confidence intervals are reported by the data-sampling-based engines during the approximate query execution, without referring to the knowledge about the exact query results treated as a baseline. Nevertheless, we intend to deliver some mechanisms producing analogous confidence intervals with respect to the expected accuracy of approximate query outcomes, i.e., intervals estimating the degrees of similarity between the observed approximate results and the "hypothetical" (not observed) exact results.

Confidence intervals might also support decisions about switching between the approximate and exact modes within the platforms such as SuperScale Analytics. As outlined in Section III, the considered framework includes two engines triggered for different types of queries. It is worth noting again that both of these engines are based on the ideas of information granulation and rough set approximations, although they are

designed for different styles of query execution [26]. Currently, the "switch" between the engines is based on the expected cardinality of data rows satisfying query conditions. Another possibility would be to rely on the assessment of a trust in an approximate query result. For instance, for a query that includes the `a = v` condition, our trust should be rather low if `v` does not occur as a special value in the summaries representing column `a` and, on the other hand, those summaries do not provide enough insight to deny the occurrence of `v` on `a` in the corresponding collections of original rows.

Going further, our approach to the data summary processing can reach beyond `SQL` [7]. Actually, the literature contains many ideas how to use granulated data summaries for the machine learning purposes [1]. As discussed in Section II, an adoption of the machine learning methods becomes crucial also in cyber-security [27]. The current design of SuperScale Analytics assumes that the machine learning and approximate query routines are kept apart. The machine learning algorithms are executed over the detailed data to train better anomaly/threat detection models, while the approximate query layer is useful for human experts to assess situations that those models could not handle. However, given the observed growth of the data, there is an emerging requirement for new machine learning tools that can work directly on summaries.

Another non-`SQL` thread of discussions in Section II referred to the area of visual analytics. Its importance is already well-recognized from the perspective of cyber-security applications [24], although it can be treated also as a broader topic within the realm of log analytics or data analytics in general. According to our research reported in [8], it is fully possible to develop data visualization and interaction libraries connecting directly to a layer of granulated summary structures.

In conclusion, the marriage of the SuperScale Analytics platform developed for specific cyber-security purposes and the summary-based database engine aimed at a general support for approximate explorative querying makes a lot of sense from both commercial and scientific perspectives. The deployment of the proposed engine within the SuperScale Analytics production environment allowed us to address the emerging needs of SoD clients. On the other hand, the real-world use case of SuperScale Analytics turned out to be truly helpful to evaluate the current engine's capabilities, to make a progress in the area of the accuracy versus performance testing, and to add some important items to the research roadmap.

REFERENCES

[1] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. On Clustering Massive Data Streams: A Summarization Paradigm. In C. C. Aggarwal, editor, *Data Streams – Models and Algorithms*, pages 9–38. Springer, 2007.

[2] A. Agrawal, J. Choi, K. Gopalakrishnan, S. Gupta, R. Nair, J. Oh, D. A. Prener, S. Shukla, V. Srinivasan, and Z. Sura. Approximate Computing: Challenges and Opportunities. In *Proc. of ICRC 2016*, pages 1–8.

[3] J. Benton, R. P. Goldman, M. H. Burstein, J. Mueller, P. Robertson, D. Cerys, A. Hoffman, and R. Bobrow. Active Perception for Cyber Intrusion Detection and Defense. In *Proc. of AAAI 2016 Workshops, vol. WS-16-03*, pages 157–164.

[4] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. *Network Traffic Anomaly Detection and Prevention – Concepts, Techniques, and Tools*. Springer, 2017.

[5] S. Cha. Comprehensive Survey on Distance/Similarity Measures Between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307, 2007.

[6] S. Chaudhuri, B. Ding, and S. Kandula. Approximate Query Processing: No Silver Bullet. In *Proc. of SIGMOD 2017*, pages 511–519.

[7] A. Chądzyńska-Krasowska, P. Betliński, and D. Ślęzak. Scalable Machine Learning with Granulated Data Summaries: A Case of Feature Selection. In *Proc. of ISMIS 2017*, pages 519–529.

[8] A. Chądzyńska-Krasowska, S. Stawicki, and D. Ślęzak. A Metadata Diagnostic Framework for a New Approximate Query Engine Working with Granulated Data Summaries. In *Proc. of IJCRS 2017 vol. 1*, pages 623–643.

[9] G. Cormode, M. N. Garofalakis, P. J. Haas, and C. Jermaine. Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches. *Foundations and Trends in Databases*, 4(1-3):1–294, 2012.

[10] A. D. D'Amico, J. R. Goodall, D. R. Tesone, and J. K. Kopylec. Visual Discovery in Computer Network Defense. *IEEE Computer Graphics and Applications*, 27(5):20–27, 2007.

[11] S. M. Emran and N. Ye. Robustness of Chi-Square and Canberra Distance Metrics for Computer Intrusion Detection. *Quality and Reliability Engineering International*, 18:19–28, 2002.

[12] D. Fisher, I. O. Popov, S. M. Drucker, and M. C. Schraefel. Trust Me, I'm Partially Right: Incremental Visualization Lets Analysts Explore Large Datasets Faster. In *Proc. of CHI 2012*, pages 1673–1682.

[13] P. Garcia-Teodoro, J. E. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez. Anomaly-based Network Intrusion Detection: Techniques, Systems and Challenges. *Computers & Security*, 28(1-2):18–28, 2009.

[14] Y. Hu, S. Sundara, and J. Srinivasan. Supporting Time-constrained SQL Queries in Oracle. In *Proc. of VLDB 2007*, pages 1207–1218.

[15] Y. E. Ioannidis and V. Poosala. Histogram-based Approximation of Set-valued Query-Answers. In *Proc. of VLDB 1999*, pages 174–185.

[16] M. Ishiguro, H. Suzuki, I. Murase, and H. Ohno. Internet Threat Detection System Using Bayesian Estimation. In *Proc. of FIRST 2004*, pages 1–5.

[17] G. Jurman, S. Riccadonna, R. Visintainer, and C. Furlanello. Algebraic Comparison of Partial Lists in Bioinformatics. *PLOS ONE*, 7(5):1–20, 2012.

[18] R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. Wiley, 2011.

[19] B. Mozafari and N. Niu. A Handbook for Building an Approximate Query Engine. *IEEE Data Engineering Bulletin*, 38(3):3–29, 2015.

[20] G. Nápoles, I. Grau, R. Falcon, R. Bello, and K. Vanhoof. A Granular Intrusion Detection System Using Rough Cognitive Networks. In R. Abielmona, R. Falcon, N. Zincir-Heywood, and H. A. Abbass, editors, *Recent Advances in Computational Intelligence in Defense and Security*, pages 169–191. Springer, 2016.

[21] M. Nauman, N. Azam, and J. Yao. A Three-Way Decision Making Approach to Malware Analysis Using Probabilistic Rough Sets. *Information Sciences*, 374:193–209, 2016.

[22] R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, 2003.

[23] M. Poess, T. Rabl, and H. Jacobsen. Analysis of TPC-DS: The First Standard Benchmark for SQL-based Big Data Systems. In *Proc. of SoCC 2017*, pages 573–585.

[24] H. Shiravi, A. Shiravi, and A. A. Ghorbani. A Survey of Visualization Systems for Network Security. *IEEE Transactions on Visualization and Computer Graphics*, 18(8):1313–1329, 2012.

[25] D. Ślęzak, R. Glick, P. Betliński, and P. Synak. A New Approximate Query Engine Based on Intelligent Capture and Fast Transformations of Granulated Data Summaries. *Journal of Intelligent Information Systems*, 2017.

[26] D. Ślęzak, P. Synak, A. Wojna, and J. Wróblewski. Two Database Related Interpretations of Rough Approximations: Data Organization and Query Execution. *Fundamenta Informaticae*, 127(1-4):445–459, 2013.

[27] R. Sommer and V. Paxson. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In *Proc. of IEEE S&P 2010*, pages 305–316.

[28] Ł. Sosnowski. Framework of Compound Object Comparators. *Intelligent Decision Technologies*, 9(4):343–363, 2015.

[29] T. Takahashi and Y. Kadobayashi. Reference Ontology for Cybersecurity Operational Information. *The Computer Journal*, 58(10):2297–2312, 2015.

[30] Y. Yao. A Triarchic Theory of Granular Computing. *Granular Computing*, 1(2):145–157, 2016.