Getting Rid of Data

TOVA MILO, School of Computer Science, Tel Aviv University

We are experiencing an amazing data-centered revolution. Incredible amounts of data are collected, integrated, and analyzed, leading to key breakthroughs in science and society. This well of knowledge, however, is at a great risk if we do not dispense with some of the data flood. First, the amount of generated data grows exponentially and already at 2020 is expected to be more than twice the available storage. Second, even disregarding storage constraints, uncontrolled data retention risks privacy and security, as recognized, e.g., by the recent EU Data Protection reform. Data disposal policies must be developed to benefit and protect organizations and individuals.

Retaining the knowledge hidden in the data while respecting storage, processing, and regulatory constraints is a great challenge. The difficulty stems from the distinct, intricate requirements entailed by each type of constraint, the scale and velocity of data, and the constantly evolving needs. While multiple data sketching, summarization, and deletion techniques were developed to address specific aspects of the problem, we are still very far from a comprehensive solution. Every organization has to battle the same tough challenges with ad hoc solutions that are application-specific and rarely sharable.

In this article, we will discuss the logical, algorithmic, and methodological foundations required for the systematic disposal of large-scale data, for constraints enforcement and for the development of applications over the retained information. In particular, we will overview relevant related work, highlighting new research challenges and potential reuse of existing techniques.

CCS Concepts: • **Information systems → Data management systems**;

Additional Key Words and Phrases: Data disposal, data retention, data management, query answering

ACM Reference format:
Tova Milo. 2019. Getting Rid of Data. *J. Data and Information Quality* 12, 1, Article 1 (November 2019), 7 pages.
https://doi.org/10.1145/3326920

1 INTRODUCTION

Over the past few years, we are experiencing an amazing data-centered revolution in almost every aspect of our lives. Incredible amounts of data are being collected, transformed, integrated, and analyzed, leading to key breakthroughs in medicine, commerce, transportation, science, and society. This data-centered revolution is fueled by the masses of data constantly generated everywhere, but at the same time is at a great risk due to the very same information flood. First, the size of our digital universe grows exponentially, and it is estimated that by the year 2020 the demand for storage will outstrip production by six zettabytes—nearly double the available storage capacity [27]. If we do not learn how to effectively dispense with some of this data, then we will simply

Author's address: T. Milo, School of Computer Science, Tel Aviv University, Tel Aviv, Israel, 6997801; email: milo@cs.tau.ac.il.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.

1936-1955/2019/11-ART1 $15.00
https://doi.org/10.1145/3326920

drown. Second, even if we disregard storage constraints, uncontrolled data collection endangers privacy and security, as recognized, e.g., by the recent EU Data Protection Regulation (GDPR) [13]. Data disposal policies must be installed to secure both organizations and individuals.

Retaining the knowledge hidden in the data while respecting storage, processing, and regulatory constraints is a great challenge. The difficulty notably stems from the distinct, intricate requirements that each of these types of constraints entails. For instance, satisfaction of storage constraints is essentially an optimization problem where one needs to determine what data may be discarded and which summary of it to retain, if needed, so data utilization is minimally harmed. In contrast, regulatory constraints tell us what information must be deleted (or kept, in certain regulations, for a specific time period [26]), and the challenge is to identify and discard/retain all the relevant data. Adhering to both types of constraints requires that the search space for the optimal solution is restricted to those that respect the regulations, and that at least one such solution must be applied to guarantee compliance. The problem is further complicated by the scale and velocity of data and the constantly evolving needs. While multiple techniques for data sketching, summarization, compression, and removal have been proposed to address specific aspects of the problem [4, 6, 9, 18, 21, 25], we are still very far from a comprehensive solution. Every single initiative has to battle, almost from scratch, the same tough challenges. The ad hoc solutions, even when successful, are application-specific and rarely sharable.

To pave the road for successful big data management, we need to develop solid scientific foundations for Web-scale data disposal. This encompasses development of a formal model that captures all the diverse facets of data disposal. This also means developing the necessary reasoning capabilities for supporting data cleaning, integration, sharing, querying, and analysis over the dynamically evolving data. We believe that such a principled approach is essential to retain knowledge of superior quality to realize the task more effectively and automatically, be able to reuse solutions, and thereby to secure the data-centered revolution that is transforming our life.

2 BENEFITS AND CHALLENGES

An important advantage of an intelligent, systematic, data disposal is the ability to better utilize the huge information pool. The realization of using this greater pool, which would otherwise not be possible due to limited storage, increases the potential for critical information gathering and for identifying new unforeseen insights. An added incentive is the potentially reduced processing costs associated with the smaller retained data. Smaller data sets often require smaller processing resources and less sophisticated tools. Last, but important, effective enforcement of privacy and data retention regulations allows meeting legal and business archival requirements, protecting both individuals and companies.

To fulfill this great potential, however, a variety of significant conceptual and technical challenges must be addressed. The first challenge is to determine which data should/may be discarded and, if allowed, what summary may be kept for the deleted items so data utilization is minimally harmed. We call this a *data disposal policy*. Conventional data sketching and summarization techniques are mostly concerned with specific query operations, but modern data analysis often employs general purpose programming languages empowered by powerful data analysis and machine learning tools. This leads to the following intriguing question: Given a dataset, a set of constraints, and an analysis workload expressed as a class of programs (the precise representation of the expected workload is a research goal), can we effectively derive a data disposal policy so the retained information is "sufficient" (in a well-specified manner) for every program in the class? Can we execute the policy efficiently?

A second, complementary challenge is to effectively evaluate queries over the retained information. This challenge stems from the fact that the deleted data may be summarized in different ways and at different granularity levels. Query answering thus involves the identification and processing of the relevant information as well as providing to users an explanation/justification for why the returned answers are estimated to be correct.

Scaling is important here. We are accustomed to a world with billions of Web pages; we must now get accustomed to a world in which the number of facts is counted in zettas. Both data disposal and query evaluation must scale to such volumes. For example, already today sensor networks collect so much data that most of it (usually based on ad hoc decision rules) gets thrown away and is not even transmitted off the sensor to the base station/database.

Finally, being typically set in a decentralized Web environment, big data management comes with all the common difficulties of large-scale data integration: assembly of information that comes from different sources, possibly replicated, and in different formats and varying quality. We should be able to support all this pipeline for the partial, retained data.

As mentioned in the Introduction, we believe it is important to design a formal model to support Web-scale data disposal. We stress, however, that the goal is not to design a universal data disposal policy (which we do not believe is possible), but rather to develop a generic "dispose by design" framework that allows to (1) declaratively express data and data usage properties, (2) model retention constraints/criteria and desired disposal strategies, (3) enforce them, and (4) efficiently evaluate queries and run data analysis processes over the retained information. In what follows, we discuss in greater detail some of the important ingredients of such a potential model/solution. In particular, we overview relevant related work, highlighting new research challenges and potential reuse of existing techniques.

3 COMBINING DATA AND META DATA

Important to this context is the notion of data provenance [8]. In a nutshell, provenance traces the source of information and the computational process it undergoes, and is critical for understanding data usage, for explaining query results, and for assessing their validity [2, 8, 15]. In our setting, provenance must further capture what data have been omitted and what kind of summary has been retained for it (if). Such meta-data will allow for effective query evaluation over the retained information and for explanation/justification of the obtained results. It will also allow keeping record of processing activities, as required by regulations such as GDPR [13].

Attaching provenance to individual data items is usually straightforward to do [15]; what is more complex and needs to be studied in our setting is what provenance should be used to annotate summarized or omitted data. The difficulty stems from the fact that common provenance models often apply only to a fairly restricted set of declarative data manipulation operators (e.g., SQL), whereas data summarization/deletion strategies are typically given as general programs, possibly empowered by machine learning (ML) tools [18, 25], for which provenance is not yet well defined. A second, related, difficulty is to propagate such meta-data through the analysis process; for example, to indicate that an analysis result is based on data originating at sources A and B that was valid on dates X and Y and where all occurrences of Z were deleted with only their average value retained. While the above is a fairly simple example, in many cases, the analysis workflow is complex [10, 11], and the results of one step are then fed as input to further analysis steps and so forth, and as we get further away down the analysis chain, we may lose track of the origin and properties of data. In turn, this means that we would be unable to correctly assess the importance/relevance of certain data items for the computation, and the currency and reliability of the analysis results. Finally, an additional prime challenge in provenance management that we must address here is that the provenance size itself may be very large [1, 14], and so effective disposal must be recursively applied to this meta-data as well.

4 SKETCHING AND SUMMARIZATION

Much of the data that we keep around is redundant and can be discarded with no harm. Common examples are old document drafts and ancient emails dealing with non-important issues. But in general, whether or not data should be kept, and in what granularity level, depends on the expected analysis workload and the regulatory constraints. For a simple example, the analysis could in principle explicitly ask to compare multiple versions of the same document, in which case discarding old (legitimate) versions of documents without retaining record of the deltas is harmful. A key challenge, thus, is to dynamically determine which data should/may be discarded and, if allowed, what summary may be kept for the deleted items so data utilization is minimally harmed. Namely, determine to dynamically determine the *data disposal policy*.

Naturally, it would be desirable to build on the existing technology when possible. A variety of data sketching/summarization has been proposed in the literature for specific tasks. For instance, in stream data processing, incoming data is summarized on the fly using powerful sketching techniques, then discarded altogether [9]. Other, more comprehensive summarization methods analyze the full dataset with techniques ranging from dimensionality reduction to compression-based data reduction methods and algorithms for clustering, data deduplication, redundancy elimination, and implementation of network (graph) summarization concepts [21, 25]. But, as previously mentioned, each of these approaches has been designed for a specific task and no single technique is guaranteed to always achieve superior results—performance depends on the type of data and its intended usage. A difficulty in assembling them together is that the summarization policies are often hard-coded and, consequently, are inflexible and difficult to combine and optimize. Thus, what is desirable here, instead, is to have a declarative framework that allows to express (a) data and data usage properties, (b) data deletion and summarization methods, as well as the resulting summary properties, and (c) privacy/retention constraints and criteria. This would serve as input for an engine that derives disposal policies that adhere to the needs (if possible), execute them, and efficiently run queries over the retained information. Some early encouraging results on the use of declarative specification for data disposal have been presented in Reference [28] and can serve as a starting point. But much more work is needed to extend these ideas to the modern big-data world of today.

5 DYNAMIC INCREMENTAL COMPUTATION

As more and more data is accumulated and experience is gained, the system should be "learning" and improving over time. This is particularly challenging due to the complex nature of big data applications. Besides the data analysis actions, such applications often include multiple data preparation steps for data understanding, cleaning, and integration [5, 7, 16, 17]. Traditional data cleaning and integration pipelines often run offline, obtaining a clean integrated database instance, over which queries are then evaluated. There is a difficulty in applying such an approach here: Due to the large scale of the data and its continuous accumulation, not all of it may be stored, and thus cleaning/integration needs to be performed using only the partial, retained information [16, 22]. Since some of the information relevant for the process may be missing or partially summarized, existing cleaning/integration algorithms need to be extended to cope with the situation. At the same time, we must make sure that the retained data (and its provenance) are rich enough to support such cleaning and integration tasks in addition to standard data analysis.

As new data comes in and new disposal policies are employed, the retained information must be incrementally maintained [16]. Also, recall that query answering here involves not only the identification and processing of the relevant information but also providing users with an explanation/justification for why the returned answers are estimated to be correct. Updates should thus also be propagated to the provenance information used to explain the origin of the different query answers. Dynamic incremental computation (taken here in a very broad sense) is unavoidable in this setting to support all activities, from data retention to cleaning, integrating, querying, and analysis.

Relevant technologies that may be harnessed to help here include the following:

Learning. Machine learning (ML), and its derivative technologies, has gained great popularity over the past few years as a vehicle for big data analysis. In our context, ML may be employed to learn data access and usage patterns and, correspondingly, to derive effective retention policies complying with a given set of regulations. A recent work [20] demonstrated how ML can be used to design effective indexing structures, which brings up the question of whether an analogous ML approach may be employed in our context to derive effective summarizations. An added challenge in our context is that the retention policies and summaries may need to be dynamically adapted as more data comes in (or is disposed of) and the data analysis workloads change.

Views and incremental view maintenance. It is interesting to observe that the retained information may be abstractly viewed as a *view* over the full (missing) data. Query evaluation over views has been extensively studied in the literature. Techniques for rewriting queries to be answered, as accurately as possible, using the views alone [12], are relevant to our context where queries may be evaluated only using the retained data. Incremental view maintenance has also been extensively studied in the literature [19], and the results are relevant here. A challenge particular to our setting, however, is that not only that the data is updated here (as more data is accumulated) but also the view definition itself (what data is retained/summarized and how) may change following the accumulated knowledge and/or shifts in the workloads. The relationship to incremental machine learning will also need to be investigated.

Human-in-the-loop. Depending on the context, data disposal may be executed automatically or may require to involve human input to approve data disposal or choose among multiple (possibly prioritized) disposal policies that the system suggests. Feedback on query results may also help to identify pitfalls in data retention and to help in improving the system's performance. Research on the use of human input in data management—in particular, in recent work on crowdsourcing [3, 23, 24]—has derived efficient methods for information gathering and utilization. The challenge in applying these results here is to effectively combine user input with the new data that is constantly accumulated and the evolving analysis workloads.

Scaling. Scaling is important here. We are accustomed to a world with billions of Web pages; we must now get accustomed to a world in which the number of facts is counted in zettas. As mentioned, already today sensor networks collect so much data that most of it gets thrown away (usually based on ad hoc decision rules) and is not even transmitted to the database. Footage from security cameras is another example where we already throw away data in ad hoc rule-based manner. Both our intelligent data disposal policies and the corresponding query evaluation algorithms must scale to such volumes.

Approximate query processing. The ability to provide approximate answers to queries, at a fraction of the cost of executing the query in the traditional way, has made approximate query processing [6] extremely popular in big data applications. Sampling is a common tool in such systems. In query-time sampling, the query is evaluated over samples taken from the database at run time. Employing such techniques in our context requires extending the sampling mechanisms to sample also from the data summaries (which provide an aggregated description of the missing data elements) and, correspondingly, incorporating such samples in query evaluation. To achieve a sharper reduction on response time, some approximate query processing algorithms draw samples from the data in a pre-processing step, then use them to process incoming queries. An intriguing question to be examined is whether such precomputed samples can themselves serve as data summaries in our context, thereby allowing to discard some (or all) of the remaining items.

6 CONCLUSION

In summary, intelligent data disposal allows companies and researchers to focus attention on valuable legitimate information, and will secure the data-centered revolution that is transforming our life. This short article highlights the corresponding research challenges and overviews some of the relevant related work and potential reuse of existing techniques. A topic that we did not discuss in the article, but deserves attention, is verification. Policies such as the recent EU General Data Protection Regulations set the legal basis for data retention and usage. The ability to verify organizations' compliance with the regulations is an important enabling factor for their enforcement. The need to perform such verification without compromising privacy and security leads to fascinating challenges, involving, e.g., techniques in encryption, differential privacy, automatic deduction, and program analysis. The data layer that we discussed in this article aims to provide support for the compliance to such regulations. Verification is a complementary exciting research direction.

REFERENCES

[1] E. Ainy, P. Bourhis, S. B. Davidson, D. Deutch, and T. Milo. 2015. Approximated summarization of data provenance. In *Proceedings of the CIKM*. 483–492.

[2] Y. Amsterdamer, S. B. Davidson, D. Deutch, T. Milo, J. Stoyanovich, and V. Tannen. 2011. Putting lipstick on pig: Enabling database-style workflow provenance. *PVLDB* 5, 4 (2011), 346–357.

[3] Yael Amsterdamer, Yael Grossman, Tova Milo, and Pierre Senellart. 2013. Crowd mining. In *Proceedings of the SIGMOD*. 241–252.

[4] M. Besta and T. Hoefler. 2018. Survey and taxonomy of lossless graph compression and space-efficient graph representations. Retrieved from *CoRR* abs/1806.01799 (2018).

[5] A. Calì, D. Calvanese, and M. Lenzerini. 2013. Data integration under integrity constraints. In *Seminal Contributions to Information Systems Engineering, 25 Years of CAiSE*. 335–352.

[6] S. Chaudhuri, B. Ding, and S. Kandula. 2017. Approximate query processing: No silver bullet. In *Proceedings of the SIGMOD*.

[7] C. Chen, B. Golshan, A. Y. Halevy, W. C. Tan, and A. Doan. 2018. BigGorilla: An open-source ecosystem for data preparation and integration. *IEEE Data Eng. Bull.* 41, 2 (2018), 10–22.

[8] J. Cheney, L. Chiticariu, and W. C. Tan. 2009. Provenance in databases: Why, how, and where. *Found. Trends Datab.* 1, 4 (2009), 379–474.

[9] G. Cormode. 2017. Data sketching. *Commun. ACM* 60, 9 (2017), 48–55.

[10] D. Deutch and T. Milo. 2012. *Business Processes: A Database Perspective*. Morgan & Claypool Publishers.

[11] D. Deutch and T. Milo. 2012. A structural/temporal query language for business processes. *J. Comput. Syst. Sci.* 78, 2 (2012), 583–609.

[12] A. Doan, A. Y. Halevy, and Z. G. Ives. 2012. *Principles of Data Integration*. Morgan Kaufmann.

[13] GDPR. (2016). General Data Protection Regulation (GDPR). Retrieved from https://en.wikipedia.org/wiki/General_Data_Protection_Regulation.

[14] B. Glavic and G. Alonso. 2009. Perm: Processing provenance and data on the same data model through query rewriting. In *Proceedings of the ICDE*. 174–185.

[15] T. J. Green and V. Tannen. 2017. The semiring framework for database provenance. In *Proceedings of the PODS*. 93–99.

[16] Ihab F. Ilyas. 2016. Effective data cleaning with continuous evaluation. *IEEE Data Eng. Bull.* 39, 2 (2016), 38–46.

[17] H. V. Jagadish, J. Gehrke, A. Labrinidis, Y. Papakonstantinou, J. M. Patel, R. Ramakrishnan, and C. Shahabi. 2014. Big data and its technical challenges. *Commun. ACM* 57, 7 (2014), 86–94.

[18] M. L. Kersten and L. Sidirourgos. 2017. A database system with amnesia. In *Proceedings of the CIDR*.

[19] C. Koch, D. Lupei, and V. Tannen. 2016. Incremental view maintenance for collection programming. In *Proceedings of the PODS*. 75–90.

[20] T. Kraska, A. Beutel, E. Chi, J. Dean, and N. Polyzotis. 2018. The case for learned index structures. In *Proceedings of the SIGMOD*. 489–504.

[21] Y. Liu, T. Safavi, A. Dighe, and D. Koutra. 2018. Graph summarization methods and applications: A survey. *ACM Comput. Surv.* 51, 3 (2018), 62:1–62:34.

[22] R. J. Miller. 2017. The future of data integration. In *Proceedings of the SIGKDD.*

[23] Tova Milo. 2017. The smart crowd—Learning from the ones who know. In *Proceedings of the ICDT.* 3:1–3:1.

[24] Aditya G. Parameswaran, Akash Das Sarma, and Vipul Venkataraman. 2016. Optimizing open-ended crowdsourcing: The next frontier in crowdsourced data management. *IEEE Data Eng. Bull.* 39, 4 (2016), 26–37.

[25] M. H. Rehman, C. S. Liew, A. Abbas, P. P. Jayaraman, T. Y. Wah, and S. U. Khan. 2016. Big data reduction methods: A survey. *Data Sci. Eng.* 1, 4 (2016), 265–284.

[26] retention [n.d.]. Data Retention. Retrieved from https://en.wikipedia.org/wiki/Data_retention.

[27] L. Rizzatti. 2016. Digital data storage is undergoing mind-boggling growth. *EETimes Magazine* (Sept. 14, 2016).

[28] J. Skyt, C. S. Jensen, and T. Bach Pedersen. 2008. Specification-based data reduction in dimensional data warehouses. *Inf. Syst.* 33, 1 (2008), 36–63.

Received March 2019; revised April 2019; accepted April 2019