EMBnet.journal
Bioinformatics in Action

# Genomic big data hitting the storage bottleneck

**Louis Papageorgiou[1,2], Picasi Eleni[1], Sofia Raftopoulou[1,3,4], Meropi Mantaiou[3], Vasileios Megalooikonomou[5], Dimitrios Vlachakis[1,4,5]**

[1] Laboratory of Genetics, Department of Biotechnology, School of Food, Biotechnology and Development, Agricultural University of Athens, Athens, Greece
[2] Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Athens, Greece
[3] Sotiria Chest Diseases Hospital, Athens, Greece
[4] Division of Endocrinology and Metabolism, Center of Clinical, Experimental Surgery and Translational Research, Biomedical Research Foundation of the Academy of Athens, Athens, Greece
[5] Computer Engineering and Informatics Department, School of Engineering, University of Patras, Patras, Greece

Competing interests: LP none; PE none; SR none; MM none; VM none; DM none

**Abstract**

During the last decades, there is a vast data explosion in bioinformatics. Big data centres are trying to face this data crisis, reaching high storage capacity levels. Although several scientific giants examine how to handle the enormous pile of information in their cupboards, the problem remains unsolved. On a daily basis, there is a massive quantity of permanent loss of extensive information due to infrastructure and storage space problems. The motivation for sequencing has fallen behind. Sometimes, the time that is spent to solve storage space problems is longer than the one dedicated to collect and analyse data. To bring sequencing to the foreground, scientists have to slide over such obstacles and find alternative ways to approach the issue of data volume. Scientific community experiences the data crisis era, where, out of the box solutions may ease the typical research workflow, until technological development meets the needs of Bioinformatics.

## Introduction

Since 1956, but mainly in the last decades, storage space needs have grown spectacularly. The problem is that, as time flows, the storage funding issue has increased more than sequencing. That is a big problem that the modern scientist has to face. Sequencing has become more troubling because this issue makes the whole procedure difficult. The motivation for sequencing and producing new data has started to fall away (De Silva and Ganegoda, 2016).

Such data comes in the form of short sequencing reads, i.e. short character strings (typically having lengths in the range 75-150). Each character represents a nucleotide (which is also called a "base"), and can assume the values of A (adenine), C (cytosine), G (guanine), T (thymine), or N (failure in the base calling process) (Langmead, 2010). The nucleotide string is usually accompanied by a corresponding string of ASCII characters, encoding the "quality" (that is, the error probability of the base calling) of each of the nucleotides. This is a representative case of how a typical sequencing setup works when a resequencing problem is considered. In such a case, a reference (possibly not 100% accurate) for the genome/transcriptome of the organism being sequenced is already known. One has to map the DNA/RNA sequence reads to the reference (*i.e.*, understand where such reads come from in the reference) and find variants present in the genetic code of the specific organisms compared to the reference (Xu *et al.*, 2014).

Depending on the biological application at hand, one might need to perform several tasks on the data, possibly in several steps, with both per-read and global computations required (Libbrecht and Noble, 2015). A typical workflow corresponding to the above use case might be as follows:

- store the reads in compressed searchable form (necessary to avoid excessive storage consumption);
- retrieve (a subset of) the reads based on some criterion, possibly depending on the experiment metadata (for instance, select all the sequencing reads derived from a given tissue subject to a specific biological condition);
- select/process the reads, for example: identify all the reads containing long stretches of low-quality nucleotides, and trim/eliminate them;
- pattern/match the surviving data, read by read, onto a reference genome;
- store the reads and their alignments to the reference genome (that is, the matches found in the genome for each read) in compressed searchable form again.

In the meantime, the Cern data centre has upgraded storage capacity on 200 petabytes, breaking the previous record of 100 petabytes. Information produced every day is one petabyte per second. This leads to lack of space

**Article history**
Received: 14 February 2018
Accepted: 07 March 2018
Published: 19 April 2018

capacity within 3 minutes. Then all this information has to be filtered for any findings which are stored for later use, after three minutes everything is deleted and three minutes is a very short period to trace back all this information (Britton and Lloyd, 2014).

All this data that need to be retrieved and handled is being held up in I/O traffic because of slow processing power (Fan *et al.*, 2014). Even if process power isn't still satisfying for such needs, there are other ways to slide over this obstacle. Technology and science go on hand by hand, and someone has to think out of the box to solve any occurring problem, without being stuck conventionally. The other suggested path is the information packings. By limiting, not only the data space needed for the information that we already have but also the new information we get, we can go further in a less chaotic and more organised environment by throwing away unnecessary information (repeats) (Fan *et al.*, 2014).

The important thing is to compress information without losing data that is needed. One should keep in mind that not only huge amounts of data will need to be processed each day, but also that some operations might need to be performed incrementally. For instance, the data produced at some point might be used to refine the results obtained from some other data generated previously, implying the reprocessing of a possibly much bigger dataset. For these reasons the development of a robust and extensible high-throughput storage/matching/processing system is necessary. Many other workflows might be envisaged, but most of them share the same skeleton structure, that is storage, retrieval, filtering/processing, and final storage of the results.

Clustering information based on a representative model (in some permissible limits) is an interesting way to approach the problem (Slonim *et al.*, 2005). For instance, when information is recorded in output, the ones that don't differ from our first recorded ones should not be referred. The differences are the essential information for our search.

To some extent, sequencing data are intrinsically noisy (they depend on chemical reactions which are stochastic in nature) (Alvarez et al., 2015). On the one other hand, high-throughput sequencing techniques have now reached a high degree of reliability, so sequencing errors are relatively rare (Pareek *et al.*, 2011). Also, as mentioned above, sequencing machines provide a quantification of the sequencing error at each nucleotide regarding “qualities”, which can be used to pinpoint problematic nucleotides/regions in the read.

## Storage state of the art

Since several years, under the pressure of increasing volumes of data and due to reduced hardware costs, the view of databases as centralised data access points has become vaguer (Sreenivasaiah and Kim, 2010). Fundamental paradigms of data organisation and storage have been revised to accommodate parallelisation, disreputability and efficiency. The storage mechanics, the querying methods and the analysis and aggregation of the results follow new models and practices. Search has gone beyond the boolean match, being directly linked to efficient indexes allowing approximate matching in domains ranging from string to graph matching (Pienta *et al.*, 2016). The main points of this progress can be summarised as follows.

From row-oriented representation, nowadays the trend is to move to column-oriented representation and database systems (Abadi *et al.*, 2009), which are the evolution of what was called “large statistical databases” in earlier literature (Corwin *et al.*, 2007; Turner *et al.*, 1979). Column-oriented database systems allow high compressibility per column (Abadi *et al.*, 2008), by direct application of existing ratio-optimised compression algorithms (Abadi *et al.*, 2006). Furthermore, several threads are pulling current database practices away from the relational paradigm. Large-scale storage and access may include dynamic control over data layout. Peer-topeer (P2P) overlays are also used in distributed stores, exchanging, e.g., index information to contributing nodes in distributed data warehouses (Doka *et al.*, 2011), where even the queries can be executed in a peerbased fashion spreading the processing load. Another alternative, related to large-scale analysis is the case of Pig Latin (Gates *et al.*, 2009), where a SQL-like syntax is used to provide the data flow requirements for analysis over a map-reduce infrastructure. Other efforts offer partial SQL support, as is the case of Hive (Ashish *et al.*, 2010) and the corresponding query language, named HiveQL.

Recently, parallel databases (*e.g.*, Oracle Exadata, Teradata) allowed high efficiency at the expense of failure recovery and elasticity (Pavlo *et al.*, 2009). Newer approaches and versions of these parallel databases integrate a map-reduce approach into the systems to alleviate these drawbacks, see (Abouzeid *et al.*, 2009) for more information.

The increased availability of low-cost, legacy computers has brought cloud computing settings to the front line. Shared-nothing architectures, implying selfsufficient storage or computation nodes, are applied to storage settings (O'Driscoll *et al.*, 2013). There exist also alternative clouds based on active data storage (Delmerico *et al.*, 2009; Fan *et al.*, 2014) where part of the computational database effort is distributed among the processing units of storage peripherals. Such an example is the case of DataLab (Moretti *et al.*, 2010) where data operations, both read and write, are based on “sets” - essentially named collections of files - distributed across several active storage units (ASUs).

Finally, task-focused storage solutions are devised to face problems in bioinformatics (Hsi-Yang Fritz *et al.*, 2011), social networks (Ruflin *et al.*, 2011) and networkmonitoring and forensics (Giura and Memon, 2010), showing how much data requirements drive the need for research on storage systems. Especially in bioinformatics, there exist approaches that combine compressed storage and indexing under a common approach, based on sequence properties and works on indexed string storage (Arroyuelo and Navarro, 2011; Ferragina and Manzini, 2005). There are cases where the system provides tunable parameters that allow a balance between data reuse and space recovery (Hsi-Yang Fritz *et al.*, 2011), by keeping only the data that may be reused shortly. At this point it must be stressed that there still exist relational databases that are used for high-throughput data storage, an example being the NCBI GEO archive (Barrett *et al.*, 2009) which supports the submission of experimental outputs and provides a set of tools to retrieve, explore and visualise data. However, even in the case of NCBI GEO, the relational nature of the underlying database is used to identify specific datasets and not specific sequences (*i.e.*, instances). Further analysis tools are used to locate sequences and aggregate information from them. In time series and sensor networks, storage can be a severe problem. In the literature, there are methods such as Sparse Indexing (Lillibridge *et al.*, 2009), where sampling and backup streams are used to create indexes that avoid disk bottlenecks and storage limitations.

Beyond the full-text indexing - combined with compressed storage, as explained above - often met in bioinformatics, there are several works on time series indexing and graph indexing. These two types of indexes, together with the string (and, thus, sequence) indexes, provide full artillery of methods that can cope with a great variety of problems and settings. Graph indexing is under massive research, due to its applicability on such cases as chemical compounds, protein interactions, XML documents, and multimedia.

Graph indexes are often based on frequent subgraphs (Yan *et al.*, 2005), or otherwise "semantically" interesting (Jiang *et al.*, 2007). There exist hierarchical graph index methods (Abello and Kotidis, 2003), and hash-based ones. A related recent work (Schafer *et al.*, 2017) relies on "fingerprints" of graphs - derived from hashing on cycles and trees within a graph - for efficient indexing. The method is part of an open source software, named "Scaffold Hunter", for visual analysis of chemical compound databases.

In the case of time series, to efficiently process and analyse large volumes of data, one must consider operating on summaries (or approximations) of these data series. Several techniques have been proposed in the literature (Anguera *et al.*, 2016), including Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Piecewise Aggregate Approximation (PAA) , Discrete Wavelet Transform (DWT) , Adaptive Piecewise Constant Approximation (APCA) , Approximation (SAX), and others. Recent works (Emil Gydesen *et al.*, 2015) based on the iSAX (Shieh and Keogh, 2009) algorithm have focused on the batch update process of indexing very large collections of time series and have proposed highly efficiency algorithms with optimised disk I/O, managing to index "one billion time series" very efficiently on a single machine. Another system, Cypress (Reeves *et al.*, 2009), applies multi-scale analysis to decompose time series and to obtain sparse representations in various domains, allowing reduced storage requirements. Furthermore, this method can answer many statistical queries without the need to reconstruct the original data.

## Conclusions

The life sciences are becoming a "big data business". Modern science needs have changed, and lack of storage space has become of great interest among the scientific community. There is an urgent need for computational ability and storage capacity development. In a short period, several scientists are finding themselves unable to extract full value from the large amounts of data becoming available. The revolution that happened in next-generation sequencing, bioinformatics and biotechnology are unprecedented. Sequencing has to come first in priority but, because of technical problems during this process, the time spent to solve space problems is longer than the one dedicated to the part of collecting and analysing data. During this problem, a huge amount of data produced every day is being lost. As we understand, the scientist must overcome some hurdles, from storing and moving data to integrate and analysing it, which will require a substantial cultural shift. Moreover, similar problems will appear in many other fields of life science. As an example, the challenges that neuroscientists have to face in the future will be even greater than those we nowadays deal with the next generation sequencing in genomics. The nervous system and the brain are far more complicated entities than the genome. Today, the whole genome of a species can fit on a CD, but in the future how we will handle the brain which is comparable to the digital content of the world. Therefore, new technological methods more effective and efficient must be found, to serve the needs of scientific search. Solving that "bottleneck" has enormous consequences for human health and the environment.

## Acknowledgements

The research reported in the present paper was partially supported by the FrailSafe Project (H2020-PHC-21-2015 - 690140) "Sensing and predictive treatment of frailty and associated co-morbidities using advanced personalized models and advanced interventions", co-funded by the European Commission under the Horizon 2020 research and innovation programme.

## References

1. Abadi D, Boncz P and Harizopoulos S (2009) Column-oriented database systems, Proceedings of the VLDB Endowment, **2**(2), 1664-1665. http://dx.doi.org/10.14778/1687553.1687625
2. Abadi D, Madden S and Ferreira M (2006) Integrating Compression and Execution in Column-Oriented Database Systems, Proceedings of the 2006 ACM SIGMOD international conference on Management of data, **1**, 671-682. https://doi.org/10.1145/1142473.1142548

3. Abadi D, Madden S and Hachem N (2008) Column-stores vs. row-stores: how different are they really?, Proceedings of the 2008 ACM SIGMOD international conference on Management of data, 67-980. http://dx.doi.org/10.1145/1376616.1376712
4. Abello J and Kotidis Y (2003) Hierarchical graph indexing. Proceedings of the twelfth international conference on Information and knowledge management. ACM, New Orleans, LA, USA, **1**, 453-460. https://doi.org/10.1145/956863.956948
5. Abouzeid A, Pawlikowski K, Abadi D, Saliberschatz A and Rasin A (2009) HadoopDB: an architectural hybrid of MapReduce and DBMS technologies for analytical workloads, Journal Proceedings of the VLDB Endowment, **2**(1), 922-933. http://dx.doi.org/10.14778/1687627.1687731
6. Alvarez J R, Skachkov D, Massey S E, Kalitsov A and Velev J P (2015) DNA/RNA transverse current sequencing: intrinsic structural noise from neighboring bases, Frontiers in genetics, **6**(213), 1-11. https://doi.org/10.3389/fgene.2015.00213
7. Anguera A, Barreiro J M, Lara J A and Lizcano D (2016) Applying data mining techniques to medical time series: an empirical case study in electroencephalography and stabilometry, Computational and structural biotechnology journal, **14**, 185-199. https://doi.org/10.1016/j.csbj.2016.05.002
8. Arroyuelo D and Navarro G (2011) Space-efficient construction of Lempel–Ziv compressed text indexes, Information and Computation, **209**(7), 1070-1102. https://doi.org/10.1016/j.ic.2011.03.001
9. Ashish T, Joydeep S, Namit J, Zheng S, Prasad C, et al. (2010) Hive A Petabyte Scale Data Warehouse Using Hadoop, Proceedings of the 26th International Conference on Data, **1**, 996-1005. http://dx.doi.org/10.1109/ICDE.2010.5447738
10. Barrett T, Troup D B, Wilhite S E, Ledoux P, Rudnev D, et al. (2009) NCBI GEO: archive for high-throughput functional genomic data, Nucleic acids research, **37**, 885-890. https://doi.org/10.1093/nar/gkn764
11. Britton D and Lloyd S L (2014) How to deal with petabytes of data: the LHC Grid project, Reports on progress in physics. Physical Society. https://doi.org/10.1088/0034-4885/77/6/065902
12. Corwin J, Silberschatz A, Miller P L and Marenco L (2007) Dynamic tables: an architecture for managing evolving, heterogeneous biomedical data in relational database management systems, Journal of the American Medical Informatics Association : JAMIA, **14**, 86-93. https://doi.org/10.1197/jamia.M2189
13. De Silva P Y and Ganegoda G U (2016) New Trends of Digital Data Storage in DNA, BioMed research international. http://dx.doi.org/10.1155/2016/8072463
14. Doka K, Tsoumakos D and Koziris N (2011) Online Querying of Dimensional Hierarchies, Journal of Parallel and Distributed Computing, **71**(3), 424-437. http://dx.doi.org/10.1016/j.jpdc.2010.10.005
15. Delmerico J A, Byrnes N A, Bruno A E, Jones M D, Gallo S M, et al. (2009) Comparing the performance of clusters, Hadoop, and Active Disks on microarray correlation computations. 2009 International Conference on High Performance Computing (HiPC), 378-387. http://dx.doi.org/10.1109/HIPC.2009.5433190
16. Emil Gydesen J, Haxholm H, Sonnich Poulsen N, Wahl S and Thiesson B (2015) HyperSAX: Fast Approximate Search of Multidimensional Data. http://dx.doi.org/10.5220/0005185201900198
17. Fan J, Han F and Liu H (2014) Challenges of Big Data Analysis, National science review, **1**, 293-314. https://doi.org/10.1093/nsr/nwt032
18. Ferragina P and Manzini G (2005) Indexing compressed text, J. ACM, **52**, 552-581. http://dx.doi.org/10.1145/1082036.1082039
19. Gates A, Natkovich O, Chopra S, Kamath P, Narayanamurthy S, et al. (2009) Building a high-level dataflow system on top of Map-Reduce: the Pig experience, Proceedings of the VLDB Endowment, **2**(2), 1414-1425. http://dx.doi.org/10.14778/1687553.1687568
20. Giura P and Memon N (2010) NetStore: An Efficient Storage Infrastructure for Network Forensics and Monitoring., International Workshop on Recent Advances in Intrusion Detection, **6307** 277-296. https://doi.org/10.1007/978-3-642-15512-3_15
21. Hsi-Yang Fritz M, Leinonen R, Cochrane G and Birney E (2011) Efficient storage of high throughput DNA sequencing data using reference-based compression, Genome research, **21** (5), 734-740. http://dx.doi.org/10.1101/gr.114819.110
22. Jiang H, Wang H, Yu P S and Zhou S (2007) GString: A Novel Approach for Efficient Search in Graph Databases. 2007 IEEE 23rd International Conference on Data Engineering. http://dx.doi.org/10.1109/ICDE.2007.367902
23. Langmead B (2010) Aligning short sequencing reads with Bowtie, Current protocols in bioinformatics. http://dx.doi.org/10.1002/0471250953.bi1107s32
24. Libbrecht M W and Noble W S (2015) Machine learning applications in genetics and genomics, Nature reviews. Genetics, **16**(6), 321-332. http://dx.doi.org/10.1038/nrg3920
25. Lillibridge M, Eshghi K, Bhagwat D, Deolalikar V, Trezise G, et al. (2009) Sparse indexing: large scale, inline deduplication using sampling and locality. Proccedings of the 7th conference on File and storage technologies. USENIX Association, San Francisco, California, 111-123.
26. Moretti C, Bui H, Hollingsworth K, Rich B, Flynn P, et al. (2010) All-Pairs: An Abstraction for Data-Intensive Computing on Campus Grids, IEEE Transactions on Parallel and Distributed Systems, **21**(1), 33-46. http://dx.doi.org/10.1109/TPDS.2009.49
27. O'Driscoll A, Daugelaite J and Sleator R D (2013) 'Big data', Hadoop and cloud computing in genomics, Journal of biomedical informatics, **46**(5), 774-781. https://doi.org/10.1016/j.jbi.2013.07.001
28. Pareek C S, Smoczynski R and Tretyn A (2011) Sequencing technologies and genome sequencing, Journal of applied genetics, **52**(4), 413-435. https://doi.org/10.1007/s13353-011-0057-x
29. Pavlo A, Paulson E, Rasin A, Abadi D, Dewitt D, et al. (2009) A comparison of approaches to large-scale data analysis, ACM SIGMOD, International Conference on Management of data. http://dx.doi.org/10.1145/1559845.1559865
30. Pienta R, Navathe S, Tamersoy A, Tong H, Endert A, et al. (2016) VISAGE: Interactive Visual Graph Querying, AVI : proceedings of the Workshop on Advanced Visual Interfaces. AVI. http://dx.doi.org/10.1145/2909132.2909246
31. Reeves G, Liu J, Nath S and Zhao F (2009) Managing massive time series streams with multi-scale compressed trickles, Proc. VLDB Endow., **2**(1), 97-108. http://dx.doi.org/10.14778/1687627.1687639
32. Ruflin N, Burkhart H and Rizzotti S (2011) Social-data storagesystems. Databases and Social Networks. ACM, Athens, Greece. http://dx.doi.org/10.1145/1996413.1996415
33. Schafer T, Kriege N, Humbeck L, Klein K, Koch O, et al. (2017) Scaffold Hunter: a comprehensive visual analytics framework for drug discovery, Journal of cheminformatics, **9**(1), 28-32. https://doi.org/10.1186/s13321-017-0213-3
34. Shieh J and Keogh E (2009) iSAX: disk-aware mining and indexing of massive time series datasets, Data Min. Knowl. Discov., **19**, 57. https://doi.org/10.1007/s10618-009-0125-6
35. Slonim N, Atwal G S, Tkacik G and Bialek W (2005) Informationbased clustering, Proceedings of the National Academy of Sciences of the United States of America, **102**(51), 18297-18302. https://doi.org/10.1073/pnas.0507432102
36. Sreenivasaiah P K and Kim D H (2010) Current trends and new challenges of databases and web applications for systems driven biological research, Frontiers in physiology. http://dx.doi.org/10.3389/fphys.2010.00147
37. Turner M, Hammond R and Cotton P (1979) A DBMS for large statistical databases, Proceeding VLDB '79 Proceedings of the fifth international conference on Very Large Data Bases, **5**, 319-327. http://dx.doi.org/10.1109/VLDB.1979.718147
38. Xu P, Zhang X, Wang X, Li J, Liu G, *et al.* (2014) Genome sequence and genetic diversity of the common carp, Cyprinus carpio, Nature genetics, **46**(11), 1212-1219. http://dx.doi.org/10.1038/ng.3098
39. Yan X, Yu P S and Han J (2005) Graph indexing based on discriminative frequent structure analysis, ACM Trans. Database Syst., **30**(4), 960-993. http://dx.doi.org/10.1145/1114244.1114248