ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 5, Issue 2, February 2018

# Compendious and Optimized Succinct Data Structures for Big Data Store

[1] Vinesh kumar, [2] Dr Amit Asthana, [3]Sunil Kumar ,[4]Dr. Sunil Kumar
[1] Research Scholar, [2] Associate Professor, [3][4]Assistant Professor
[1][2] S.V. Subharti University, Meerut, [3] Vardhman College, Bijnor, [4] IIMT University,Meerut

***Abstract* - Data Representation in memory is one of the tasks in Big Data. Data structures include several types of tree data structures through the system can access accurate and efficient data in Big Data. Succinct data structures can play important role in data representation while data is processed in RAM memory for Big Data. Choosing a data structure for Data representation is a very difficult problem in Big Data. We proposed some solution of problems of data representation in Big Data. Data mining in Big Data can be utilized to take a decision by Data processing. We know the functions and rules for query processing. We have to either change method of data processing or we can change the way of data representation in memory. In this paper, different kind of tree data structures is presented for data representation in RAM of a computer system for Big Data by using succinct data structures. Data mining is often required in Big Data. Data must be processed in parallel or steaming manner. In this paper, we first compare all data structures by the table and then we proposed succinct data structures those are very popular now. Each tree presented for Data representation has different time and space complexities.**

**Keywords: SDS (Succinct data structures), Trees, Big Data, CDS (concurrent data structures).**

## 1. INTRODUCTION

The main data structure used in Big Data is tree. Quad tree is used Graphics and Spatial data in main memory. Sub linear Algorithms are used to handle Quad tree which is inefficient. Optimized SDS can improve functionality of different SDS like Dynamic tree, succinct tree, and Suffix tree, rank and select, FM index .Geometric data, Proteins data base, Gnome data, DNA data are large data bases for main memory. An efficient and simple representation is required in main memory of computer system. The Compressed demonstration of data has been a primary requirement nearly in the field of Computer Science for a long way. However overall quantity of storing area is not a vital problem in recent times, considering the fact that external memory can store large quantity of data and may be inexpensive, time needed to get access to information is a vital blockage in numerous programs. Right to use to outside memory has been conventionally lower than accesses to main memory, which has caused examine of recent compressed demonstrations of information which might be capable to save identical data in reduced area. Succinct data structures may include Range Minimum query, Dynamic bit vector, Suffix tree, Suffix array, Dynamic tries, DFUD etc. Bit vector and Wavelet can represent protein data base.

## 2. RELATED WORK

Hassle of data proliferation is stimulating our capability to manipulate data. Standard algorithms such as greedy in terms of space utilization and not only access a simplest part of information. The investigators noticed them and gave evidence by recent troubles in streaming of data [7] and sub linear algorithms [8]. Dissimilar these instances, various troubles need complete dataset to be saved in compressed format however require it to be enquired rapidly. In real world, compression may have a greater a long far-reaching effect than simply storing data concisely: we are able to know, and that which we will understand we are able to calculate," as detected in [9].

The Researchers have taken into consideration those troubles in numerous algorithmic contexts, which contain scheme of capable algorithms for handling highly-compressible data structures. They prudently deliberate exact resources required to signify Dynamic tree, graph [20], sequences ,dictionary [8, 9, 2, 1, 2], permutations, features [2, 18], and textual content structures indexing [5, 6, 7, 8, 9, 10].Our Future Purpose for plan in writing pseudo code with strong time and space complexity. Nevertheless, Kolmogorov complexity is not decided yet for arbitrary data, so some compression technique is known to be suboptimal in this sense. The Researchers have taken into consideration those troubles in numerous algorithmic contexts, which contain scheme of capable algorithms for handling highly-compressible data structures. Dekel has shown SDS for nearest color node [18]. Yambin completed Succinct and practical greedy embedding for geometric routing[17]. Rudolph did his work on succinctness and tractability of closure operator representations. Jose design parallel construction of succinct tree[24].

## 3 . DIFFERENT TREES AND SDS

***Table 1: Comparison of Trees and SDS for processing and Indexing in Big Data with Applications***

| Data Structures | Data base query | Complexity | Data Type | Applications |
|---|---|---|---|---|
| B-tree | Point query | O(log n) | Linear data | Apple file system, NTFS,LINUX |
| B+ tree | Point query | O(log n) | Linear data | DBMS |
| B* tree | Point query | O(log n) | Linear data | File system |
| UB-tree | Point and range query | O(logn) for linear data | Linear and MD data | Range |
| H-tree | Point query | O(log n) | Linear data | LINUX |
| Compact B-tree | Point query | O(log n) | Linear data | As of B-tree but more efficient |
| R-tree | Range query | O(log n) | MD Data | Real world Application (GPS) |
| R+ tree | Range query | O(log n) | MD Data | As of R tree |
| R* tree | Range query | Little bit more than R | Spatial data | Formation of spatial data base |
| X-tree | Range query | Worst case O(n) | MD Data | High dimension data |
| M-tree | k-NN query | Worst case O(n) | Spatial data | Accessing Spatial data |
| Hilbert R-tree | Search query | 28% less than R | MD Data | Cart graph |
| BR-tree | Point, Range, bound query | O(≤ log n) | MD Data | Distributed Data base |
| QR+ tree | Range query | Not redundant | Large scale spatial data | GIS |
| Suffix tree | Search query | O(\|p\|/B+log Bn), O(mlogBn) | Linear data/MD data | search for a pattern matching, disk accesses |
| Range tree | Range query | O(logn [+k]) | Linear data/MD data | Can be used search for a pattern matching in Big Data |
| Normal trie | Search query | O(s) where s is the length of the longest prefix | Linear data/MD data | Can be used search in Big Data |
| Succinct tree | k-NN query | $2n + o(n)$ bits and carry operations in constant time | Linear data/MD data | Can be used in Big Data |
| Dynamic tree | k-NN query | O($nm$ log$n$) | Linear data/MD data | Can be used in Big Data |
| K2 tree | k-NN query | Efficient | Linear data/MD data | Can be used in Big Data |
| Wavelet tree | k-NN query | N + o(n) bits | Linear data/MD data | Big Data representation |

Figure 1 represent different trees and succinct data structures with their time complexity and application in real world data. Here some SDS can be used in Big Data representation.

## 4. DATA STRUCTURE FOR WEATHER FORECASTING-A BIG DATA

Data structures used for Big Data with respect to a special case of weather forecasting is tree. Today the Big Data

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 2, February 2018

has become a buzz word, and still in developing stage. Weather forecasting, basically the problem of initial value, is considered by researcher as a case of Big Data, which will help to improve the accuracy of forecasting. For handling this huge data need for weather forecasting, there is a requirement of a well-organized data structure.
Through this section researcher discuss review of Big Data and role of Big Data in weather forecasting, review of Data structures used for Big Data as well as weather forecasting. Numerical Weather Prediction (NWP) is the desirable technique for weather forecasting. The data structures available till now has some limitations to apply for weather data, hence researcher plan to design a new data structure which will store the weather data efficiently.
Since from the time when the cultivation had underway we are attentive in knowing about weather deviations. Diverse approaches were established to forecast weather deviations, some were intuition based while some were scientific. Constantly user looks for accuracy of forecast. This segment deliberates improvement of weather forecasting techniques and Numerical Weather Prediction as a scientific and mathematical technique of weather forecasting.

### *4.1 SDS versus Compressed Data Structures*

*Table -2:*

| S. No. | Data structures | Features |
|---|---|---|
| 1 | Compendious Data structures (SDS) | • It consumes space near to data theoretical lower bound.<br>• It can provide algorithm for direction finding, addition and deletion search procedures.<br>• Developed by Jacob. trees, bit vectors can be coded.<br>• $2n + O(n)$Bits are utilized to denote $n$ node random binary tree. |
| 2 | Implicit Data Structures | • It is an arrangement of data that utilizes low space besides actual data elements.<br>• It is known as implicit due to most of arrangement of elements is conveyed implicitly by their command.<br>• Effective for Space<br>• It can mean $O(1)$ to $O(log\ n)$additional space.<br>• Deliberate to enhance main memory uses<br>• For instance:- Heap and Beep |
| 3 | Dense or Compreesed DataStructures | • These data structures denotes that data structures whose procedures are faster as conventional data structure but whose size can be substantially smaller.<br>• Being used on data entropy of data being signified.<br>• Suffix Array and FM- |

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 2, February 2018

| | | |
|---|---|---|
| | | index are examples that indicate for pattern matching. |
| 4 | Data Structures for search | • Permit recovery of data item from sequence of objects, for instance explicit row from a database.<br>• Permit quick access.<br>• Inadequate in retrieval of few particular types. |
| 5 | PDS(Persistent DS) | • It preserves past version of it when it is amended?<br>• Such as cons-based list.<br>• Numerous mutual references relayed data structures. |
| 6. | CDS | • A specific method for keeping or establishing to access by numerous calculating thread son a computer.<br>• Utilized in computer architectures with Multiprocessors.<br>• Can be find in abstract environment storage modules. |

4.3 Normal Trie Representations

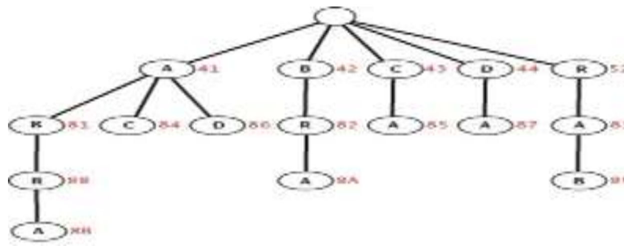

***Figure 1: Trie representation of LZW code***

Figure 2 shows the tries representation of Lempel- Ziv - Welch code. In this figure each node is in circular doubly linked list has points towards first-child and next-sibling and points to parent. It can have memory: 3 pointers (192 bits) per node. Child: time. Each node has array of σ pointers, one to each possible child. Space complexity: pointer per internal node .Child: O(1) time.

***4.4 Succinct Tries***

***Figure 2: Succinct Tries representation***

Figure 4 shows the succinct tree representation that produce the following output.
1 1111 1111 1111 1011 1110 1101 1001 0000 0011 0000 1111 0010 1111 1001 1101 1100 00111101 1011 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 00000000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

***4.5 Dynamic Tries***

In Figure 3 Dynamic trie is represented as ADT; parent(x); child(x, c); add(x, c) and Bonsai tree . Data structure: open hash table of $(1+)n$ entries; nodes of trie reside in hash table ID of a node: location where it resides; ID of child labeled c of x: Create key $hx, c\ i$ and insert. Hash table entries only store “quotients”, require only $log2\,\sigma + O\,(1)$ bits .Space usage

$(1 + )n\ log2\ \sigma\ +\ O\ (n)\ bits, O\ (1)$ time. Fast in practice (2-3 times slower than TST).

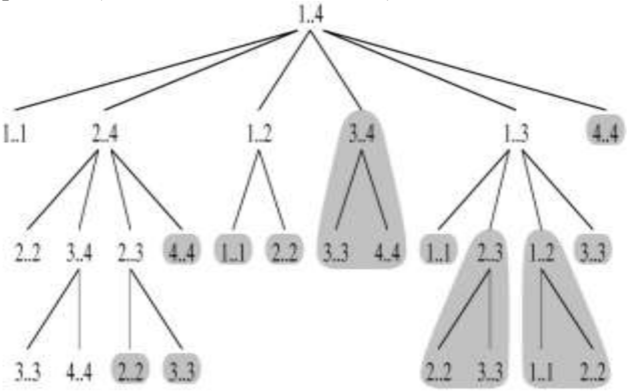


*Figure 3: Dynamic Tree as Abstract Data Type*

***4.2 Time and Space for Data***

Queries P: 10,454,552 searching queries those are from the Google query log [20]. This dataset is representative of the style and frequency of queries users may enter into the search box of a search engine or large website.

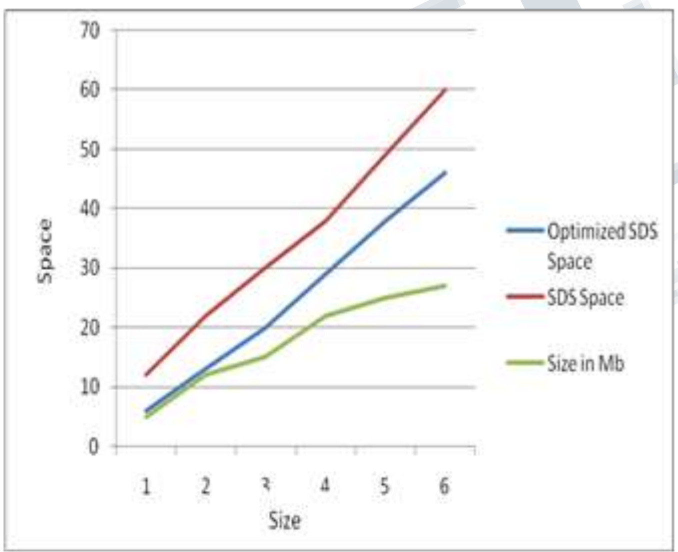


*Figure 4: Space complexity for Big Data*

Queries Q: We have filtered more than 300M search queries from Google search engine for scalability evaluation. Figure 4 and 5 shows the graphical representation of space complexity of SDS and Optimized SDS in comparison with size of stored Big Data. From below graph it is clear that Optimized SDS take very less space as compare to SDS. Figure 5 is showing performance of SDS and optimized SDS with data set from Google[20].

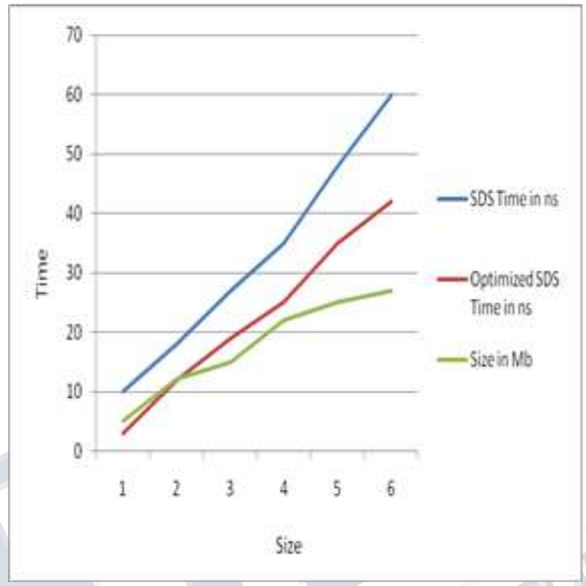


*Figure 5: Time requirement by SDS*

## 5. APPLICATIONS

SDS is applicable in area of Information retrieval for modern applications in computing devices for storing and retrieving data. It can provide support in NGS: Bowtie read aligner. Xml can use Representing XML data for internet: XML DOM "SiXML" project, having less space. It can provide Data store for Query processor in ZORBA. Many data mining tasks in on line Analyzing and Processing.

## 6. SUCCINCT LIBRARY

A SDS structures pronounced in this phase, which is up-to-date are amongst most well-organized, are agree as a part of the succinct library [12]. Library is accessible with a non-judgmental license, in hope that it will likely be beneficial both in investigation and requests. While similar in ability to other current C++ libraries for example, SDSL [1], simongog/sdl-lite and Sux [3], we completed significantly architectural choices, which we explain beneath Memory mapping. ot/succinct is library for implementation for SDS. Documentation of this library is underway in LINUX and Mac OS X .We can also tested our code here.GIT hub is web developed for succinct data structures. Succinct<T> is .NET library which adds no of features and functions to SDS. LIBCDS is library. All are open sources and available at Git web site.

## 7. CONCLUSION

Optimized SDS has capability to reduce space requirements and can handle large amount of data. SDS and optimized SDS can present highly scalable and accurate result. Implementation of SDS is little bit complex in programming language. SDS can do operations on Big Data very efficiently. Real performance can be experienced after implementation. Optimizes SDS are less time consuming but they are not easier to use practically. SDS does not support ADT fully but after optimization they can support. In futures SDS can be implemented as several libraries are developing functions and procedures for SDS.

## REFERENCES

1. G. Jacobson. Space-efficient static trees and graphs. In FOCS, pages 549–554, 1989.

2. J. I. Munro. Tables. In FSTTCS, pages 37–42, 1996.

3. R. Grossi, A. Gupta, and J. S. Vitter. High-order entropy-compressed text indexes. In SODA, pages 841–850, 2003.

4. G. Gottlob and T. Schwentick. Rewriting ontological queries into small nonrecursive datalog programs. In KR, 2012.

5. R. Grossi, A. Gupta, and J. Vitter. High-order entropy-compressed text indexes. In Proc. 14th Symposium on Discrete Algorithms (SODA), pages 841–850, 2003

6. Ladra. Algorithms and Compressed Data Structures for Information Retrieval. PhD thesis, University of A Coruña, 2011.

7. S. Muthukrishnan. Data streams: Algorithms and applications, 2003. Plenary talk at the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2003).

8. Bernard Chazelle. Who says you have to look at the input? The brave new world of sublinear computing, 2004. Plenary talk at at the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2004).

9. Scott Aaronson. NP-complete problems and physical reality. SIGACT News, 36(1):30, 2005.

10. Eric Baum. What is Thought? MIT Press, 2004.

11. David Benoit, Erik D. Demaine, J. Ian Munro, Rajeev Raman, Venkatesh Raman, and Srinivasa Rao. Representing trees of higher degree. Algorithmica, 43(4):275{292, 2005.

12. Richard F. Geary, Rajeev Raman, and Venkatesh Raman. Succinct ordinal trees with level-ancestor queries. In SODA '04: Proceedings of the _fiffteenth annual ACM-SIAM symposium on Discrete algorithms, pages 1{10. Society for Industrial and Applied Mathematics, 2004.

13. Sunil Kumar and Varun Sharma, “Working of Cloud Architecture and Components”, In Proceedings of International Multi Track Conference on Sciences, Engineering & Technical Innovations(IMTC-14), June3-4,2014, Vol.1, pp.313-316, ISBN: 978-81-929077-0-3.

14 .Gupta, A., Hon, W.K., Shah, R., Vitter, J.S.: Compressed data structures: Dictionaries and data-aware measures. In: Proc. of the 2006 IEEE Data Compression Conference (DCC ’06). (2006)

15. Yanbin Sun, Yu Zhang, Binxing Fang, Hongli Zhang Succinct and practical greedy embedding for geometric routing Computer Communications, Volume 114, 1 December 2017, Pages 51-61

16. Dekel Tsur Succinct data structures for nearest colored node in a tree Information Processing Letters, Volume 132, April 2018, Pages 6-10

17. Sebastian Rudolph Succinctness and tractability of closure operator representations Theoretical Computer Science, Volume 658, Part B, 7 January 2017, Pages 327-345

18. www.Google .com

19. Piyush Mishra and Charul Bhatanagar, "Automated sub-retinal fluid detection comprising RPE-region using neighbouring pixel connectivity paradigm", Biomedical Research, Special issue on Computational Life Sciences and Smarter Technological Advancement, pp. 1-5, 2017. Scopus (Elesvier)

20. H. Kaur and A. S. Sharma, "A Dealing with Interdependency among NFR using ISM", Pertanika Journal of Science & Technology Vol. 25, No. 3, pp. 871 - 890, 2017. (Scopus Indexed)
21. A. Sharma, A. Sharma and A.S. Jalal, "Hybrid Algorithm of Density based Clustering and Profit maximization for Facility Location Problem" in International Journal of Future Generation Communication and Networking, Vol. 10, No. 11, pp. 47-54, 2017. ESCI

22. José Fuentes-Sepúlveda, Leo Ferres, Meng He, Norbert Zeh Parallel construction of succinct trees Research article Theoretical Computer Science, Volume 700, 14 November 2017, Pages 1-22.

23. Sunil Kumar et al, "Agent based security model for Cloud Big Data", In Proceedings of 2nd International Conference on Information and Communication Technology for Competitive Strategies (ICTCS-2016), March 04-05, 2016, Udaipur, India, ACM, ISBN: 978-1-4503-3962-9, DOI: http://dx.doi.org/10.1145/2905055.2905202.

24. Sunil Kumar et al, "Authentication and encryption in cloud computing", In Proceedings of 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials(ICSTM), IEEE Xplore, 06-08 May,2015, Chennai. pp.216-219, ISBN: 978-1-4-4799-9854-8, DOI: 10.1109/ICSTM.2015.7225417

IFERP connecting engineers... developing research