Graph Based Indexing Techniques for Big Data Analytics: a Systematic Survey

V.Thiruppathy Kesavan, B.Santhosh Kumar

Abstract: ***Big data is a process which is used when the insights and meaning of stored data cannot be discovered with the existing data mining and handling techniques. The relational database engines cannot process very large datasets or an unstructured data. The relational database engines cannot process very large datasets or an unstructured data. This large size of data needs a distinguished way of processing approach which is called big data. Big data applies parallelism on the available hardware devices. Frequent changes on things make frequent changes on captured and recorded data. Specifically, the data creation, storage, retrieval and analysis is called big data, which is an enormous amount of data with respect to volume, velocity and variety. Due to the usage of devices such as mobiles, software logs, cameras, microphones and wireless sensors networks, there exist rapid growth in datasets. Hence, for efficient management and retrieval of big data, this paper investigates and examines the graph based indexing techniques for big data analysis. For storing and representing the data, a graph database is used along with the graph structures for logical requests with nodes, edges and properties. Consequently, as the datasets grow rapidly, this large set of data repositories cannot be retrieved and analyzed by using the traditional SQL model and also the relationships between the different datasets cannot be understood. In such case, the graph databases are one part of the solutions. The graph database model is obtained by extracting the relationships among different nodes or data points. It focuses in organizing and analyzing the messy data points according to the relationships, instead of looking at the value of data points. This helps in adding another layer of structuring and analyzing the data and increasing the effectiveness of big data analytics.***

Index Terms: Big data, Data mining, Indexing, Graph database

I. INTRODUCTION

The big data management challenges mainly on volume, variety and velocity of data[57,64]. First, the magnitude of a particular a data is referred as volume and in big data the magnitude or volume of data is measured and stored in terms of terabytes and petabytes, where 1 petabyte is equal to 1024 terabytes. Second challenge is the variety which depends on the type of data stored, which means the data can be structured, unstructured [1] and semi-structured. The structured data [2] involves the data on spread sheets or on relational databases, unstructured data [3] supports text, image, audio and video storages and XML is an example for semi-structured data which does not stick to any standards[63]. Third challenge of big data is its velocity which depends mainly on the rate and speed

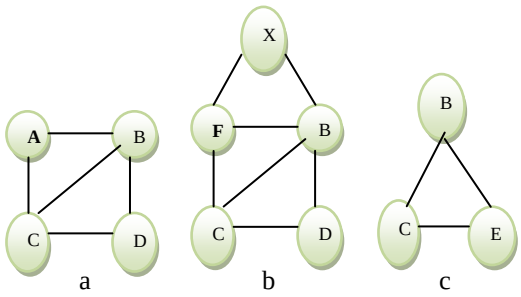


Figure 1 Sample Graph Database

of the data. For example, smart phones [4], social networks [5], e-science [6], health care systems [7] etc. generates unpredictable rate of data in every seconds. This huge rate of data generated and the speed of analyzing and retrieving those data is impossible to handle with the traditional data management systems [8].

For overcoming the above said challenges, a real time analytics is required which effects big data analytics in to existence. The technologies in big data convert the above said challenges in to opportunities and also decision making [9] plays a major role in big data analytics. Efficient decision making helps in obtaining timely responses to queries and efficient utilization of computing resources helps in improving the performance of search operations on big data. Hence, the researchers are in a situation to propose efficient data analytics solutions for the massive amount of data generated.

Technologies in big data involves CouchDB for mining the stored data [10], text mining [11] and data mining[12] etc. For producing better solutions the above said technologies are integrated with effective indexing frame work. For example, a big data processing frame work called Hadoop has improved in its task execution when it was implemented along with indexing.

Revised Manuscript Received on December 22, 2018.

V.Thiruppathy Kesavan, Professor, Department of Computer Science &GMR Institute of Technology, Razam-532127

B.Santhosh Kumar, Sr. Assistant Professor, Department of Computer Science &GMR Institute of Technology, Razam-532127, b.santhoshkumar@gmail.com

Indexing [13] can be defined as a way to quickly locate and retrieve the data along with minimum disk access and optimum performance efficiency[58-60]. However the indexing techniques that are applied to the traditional datasets and proved their efficiency are failed to prove when they are applied for big data. Hence, the researchers are forced to perform enhancements in big data indexing techniques.

Consequently, this paper investigates and examines the graph databases along with their indexing techniques which focus on minimizing the indexing overhead which in turn helps to improve the query execution and search performance for big data repositories. A graph database represents the semantic queries in graph structures with a set of nodes and edges along with its properties and relationships for storing and representing the data[61,62]. A perfect example for a graph database is Twitter, which connects 330 million active users per month. Graph databases are based on NOSQL databases which are created for overcoming the limitations of the relational databases. Below Figure 1 represents a sample graph database in which the data is stored in a graph structure which constitutes nodes, edges and their relationships and Figure 2 represents the sample possible graph queries that can be processed by the sample graph database in Figure 1.

Rest of this paper is organized as follows. Section 2 explains the existing indexing techniques on big data research community which discusses about the search cost, resource utilization, accuracy and performance efficiency. Section 3 involves a detailed survey on graph based indexing techniques with their unique features which helps in understanding the application of indexing in graph databases along with a brief tabulation. Section 4 concludes the survey.

II. INDEXING TECHNIQUES IN RESEARCH COMMUNITY

This section brings out a detailed learning on some of the currently existing data indexing techniques which discusses the search cost, resource utilization, accuracy and performance efficiency. A technique on hash based indexing [14] proved its efficiency in search performance with high-dimensional data.

A. B-Tree and Hybrid B-Tree based indexing

An extended tree of the Steiner tree called Compact Steiner Tree (CST) [15] is proposed on relational databases to improve keyword searched. Big data often deals with record of varying lengths [16] which produces multi-dimensional big data, where B-tree indexing helps in mapping related data with search keys. A set of gradually decreasing temporal data [17] values with a set of p-samples based on B-tree indexing takes linear time in sending query responses with optimum cost and for online data streams where the behaviors are unpredictable, the method attains high cost in query execution and consumes more number of computing resources. To overcome these drawbacks, to improve the efficiency of online indexing and to reduce the time complexity, a hybrid B-tree [18] indexing is proposed which mainly focus on robustness and flexibility. This method integrates a series of B+ -tree local indexes with graph partitioning.

B. Indexing on temporal networks

To A sub graph pattern matching technique [19] is proposed on temporal networks, where each time a network encounters a link the edges on the sub graph are marked with a time stamp. This network accounts edge ordering which can be accomplished through an approach called chronologically sorted edge driven approach that supports searching only on edges that occur in sequential order which results in high efficiency. For each edges, the number of matching temporal sub graphs and the time for matching the temporal sub graphs with the chronological edge driven algorithm in seconds are calculated. The main difference between a temporal and a static sub graph matching is, comparing with the number of static graphs, the number of matching temporal graphs can be either greater or lesser. Here the matching is not done on the nodes, instead it is done against the edges that follows the correct chronological ordering. Improvements on their algorithms enables high speed performance. For retrieving skewed spatial data, an R+ -tree [20] is proposed with KR+ -index which takes cloud data management(CDM) in to account. This technique is proved to be more efficient in accessing big data as well as supports nearest-neighbor queries in addition with range queries.

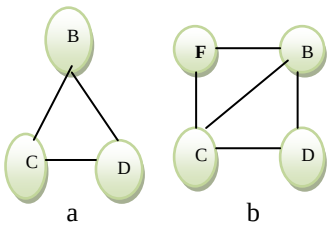


Figure 2 Graph Queries

C. Bitmap and hash based indexing

The top-k CS tree is proposed [15] which mainly focus in reducing the cost of implementing Steiner trees in searching keywords on RDBMS. Comparing with other approaches the average elapsed time is less and this method achieves high accuracy. A sequence of bits stored as bulk index data describes a bitmap indexing technique [21] is one of the efficient indexing method, where the query processing is done with bitwise logical operations. Binary encoding schemes help in constructing a bit-sliced index by dividing a set of identifiers in to number of components [22] and yet failed to give responses to queries in minimum time.

High dimensional data search introduces hash-based indexing techniques which works faster in finding similar(data duplication) data and mainly applied in real time applications like duplicate detection [23], image retrieval and document analysis. The main principle of hashing is deriving compact binary codes for every high dimensional data for obtaining better search results.A non-negative sparse coding technique [14] is proposed to convert the original feature data space in to low-dimensional data space and similarity searches are done by sparse hashing(SH) to generate binary codes. In cloud, the search process on big data can be enhanced by using semantic-based indexing [24] which uses ontology evolution to develop a cloud service environment through which the cloud resources are distributed according to the user requirements [25].

D. Fuzzy based indexing

Fuzzy rule base can be applied in indexing procedures but it is practically impossible when applied on big data with unknown events. To overcome this drawback it is recommended to use dynamic fuzzy rule tuning with hybrid fuzzy classifiers by adjusting the rules dynamically to deduce detection rate [26]. Fuzzy indexing can be applied on a huge number of moving objects and index can be created in sub-seconds [27]. To improve indexing performance, the fuzzy indexing is able to capture images at a high frame rate and this image helps in answering the queries in a short span of time. Every time a new index image is created, the previously exploited index image is discarded to free the memory space and this makes efficient memory utilization.

Fuzzy indexing is mainly used in supporting predictive queries, producing high query response time and always produces high performance rate. Collaborative indexing [28] and knowledge exploration uses a social learning model for exploring data variety. This indexing technique supports with enhanced set of semantics representation and makes them easily accessible for users who are seeking various information.

III GRAPH BASED INDEXING TECHNIQUES

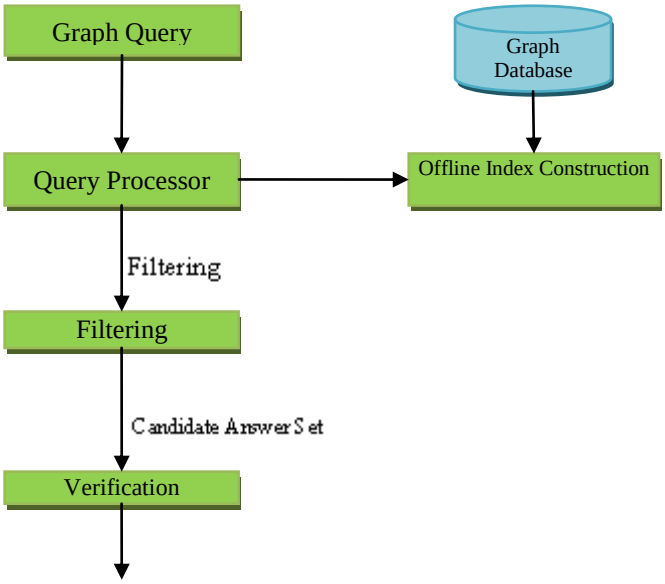
An index must be created in query processing to address the following challenges such as efficiently constructing a large database, efficiently maintaining dynamic updates and efficiently processing bulk query workloads. Indexing can also benefited by batch processing which helps in improving query processing; especially when the query graphs utilizes commonality. Figure 3 illustrates query processing with a graph database.

First, the commonalities among the stored graph data like frequent sub-graphs, paths or tress are extracted from the graph database for constructing the index. Data mining techniques like sub graph mining can be used in extracting common sub-structures from the graph database. Second is the filtering and verification phase, where the index is used to filter the false answers to obtain a candidate set and to verify every candidate whether it has any sub-graph of the received query. Figure 4 illustrates efficient graph based indexing techniques which works on graph databases that comprises a set of relative smaller graphs or sub-graphs. Each technique uses different tree based structures along with effective indexing as surveyed below.

A. Graph indexing based query processor

A graph query processing based graph indexing is proposed [29] which uses IG (Integrated Graph) query on transaction graph databases to process super graph queries. Similarity among the graphs is extracted and an index is created by using IG query. Unique features of IG Query involves, the index construction and maintenance cost is less, method of direct inclusion and filtering supports in fast query processing and batch processing [31]. Direct inclusion does not allow candidate verification instead directly include partial query answers. The candidate set is reduced by the filtering technique which is applied on a very small projected database.

Figure 3 Query processing in graph database



The process of matching graph patterns can be improved by proposing indexing graph patterns [30] for Neo4i GDB engine. Multiple indexes are created, updated and used for various graph patterns. The index is created and stored as a tree structure and stored in the same database where the base data is stored. Using indexing patterns results in high performance query, processing. Experimentation is done with three graph data sets such as a triangle index on a social graph, a funnel index on a music database and a rhombus index on a transaction database. Social graph database represents the people along with their relationships in which the nodes represent the people and the labels distinguish the males and the females. Music database stores the artists' information in which the nodes represent recorded tracks and the labels represent their record release. Transaction database stores the bank

transactions in which the nodes represent bank account numbers and the relationships are the transaction between the accounts.

The aggregation queries are authenticated [32] by introducing efficient index structures over large data sets. Considering the static environments, for achieving better performance characteristics an index is designed and for solving environments more index structures are proposed. Multiple numbers of aggregate attributes and multiple number of selection predicates results in performance excellence in authenticating queries. In addition, types of aggregate include distributive aggregates, algebraic aggregates and holistic aggregates. Even when the database is encrypted, efficient authentication on aggregation queries is achieved which supports data confidentiality.

B. A novel structure aware index

A type of Steiner tree which is more compact and meaningful is identified to study keyword search problems [15] in relational databases. To be efficient, a concept of CST(Compact Steiner Tree) [33] is proposed which produces an approximate solution for the given top-k queries [34,35]. Initially, Steiner trees [36] with minimum path weight are constructed which can be easily reconstructed using CS Trees. Furthermore, a novel structure aware index is created where the structural relationships and ranking scores are included which helps in improving the result quality and search efficiency. Any existing RDBMS can incorporate this novel based indexing technique without modifying the source code. A detailed investigation on the existing indexing techniques is done [37] to review various indexing techniques along with learning their potential and utilization in solving big data issues. Learning this paper also helps in choosing a best suit indexing technique for a specific problem. Big data indexing is broadly categorized in to Artificial Intelligence (AI) and Non-Artificial Intelligence (NAI) approaches. To detect unknown behavior in big data AI indexing approach is used. Latent Semantic Indexing (LSI) is one of the AI indexing techniques which retrieve the information by identifying the patterns between unstructured data sets. LSI works semantically over the data sets and identifies the common contexts and establishes the relationships. Hidden Markov Model (HMM) is another AI indexing approach which uses pattern recognition and the data relationship. Based on the current state, the query results are predicted for the future states. NAI approach creates an index with items which are most often searched in a particular data set. This paper covers B-tree, R-tree, inverted indexing approach and custom indexing for learning NAI approach.

C. KR+ index and Secondary Indexes

Based on the existing cloud data management (CDM) [38], a scalable and multi-dimensional index called KR+ -index is proposed [20] by assigning key names for all R+ -tree leaves. First, R+ -tree [40] is used in data dividing and the tree index leaf nodes which are rectangles is considered as dynamic grids. R+ -tree is good, because by adjusting the parameters we can balance between the size of the grid and the grid access time. Also, R+ -tree avoids overlapping of leaf nodes and thus redundant retrieval of similar data is not possible and so each rectangle of a leaf node [41] can be defined with different keys. In addition, a spatial query algorithm [42] is redefined which includes range query and k-NN query.

A taxonomy of NOSQL secondary indexes [43] is presented in this paper which supports with the guidelines in choosing secondary indexes and this is splitted in to two classes: Embedded Indexes and Stand Alone Indexes. Inside the primary table light weight filters are embedded and called Embedded Indexes which offers high throughput with more space efficiency and Stand Alone Indexes are implemented using separate data structures which results in faster query response. For LSM based NOSQL storages, secondary indexing methods like Eager, Lazy and Composite, Zone Maps and Bloom filters are studied. Level DB++ system is developed and these indexing methods are implemented on top of Level database which results in experimenting the trade-offs between these indexing techniques. A graph based hybrid spatio-temporal indexing technique [39] is proposed for dynamic multi-modal scene data storage and retrieval. An index is constructed for the task oriented scene data and an optimization algorithm is applied on the spatio- temporal relation graph index. Index generation is made excellent with the use of spatio temporal relation graph, which performs well in solving complex and uncertain spatio temporal queries. The proposed spatio temporal relation graph index method focus on collecting and combining the time series data and associated data for performing visualization tasks. This algorithm works with a graph base d index to represent various aspects of multi modal spatio temporal entities along with time, location, relation and semantic representations.

D. R-Tree and B-Tree index

Retrieving top-k query [44] for both location-aware and region-aware by an indexing technique is proposed [45]. A novel based method is defined which tightly integrates the inverted file for retrieving text and the R-tree for querying in spatial databases [46]. This method comprises algorithms that make use of the proposed index in evaluating the top-k query [47] and also encompasses both spatial proximity [48] and text relevancy to reduce the search space while processing the query.

A composite tree index structure [54] is proposed to deal with event searching for multiple keyword based queries in which bi-directional references are added between the leaf nodes and the event indices which results in less number of CPU processing cycles in comparing the data. A data structure called composite tree index build the solution in which all leaf nodes in a B-tree shares a common list of event indices.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

To process time-stamp based query efficiently, search index data structure is allowed. Furthermore, this proposed solution produces quicker response time than the traditional query processing methods.

E. Lindex and Network structured index

A graph index called Lindex is proposed [49] which comprise all the sub graphs of the database graphs. Key-Value pairs are represented by the nodes in Lindex in which key represents the sub graph in the database and the value represents the list of database graphs which contains the key. Lindex is used in improving the efficiency of sub

Author	Technique / Algorithm	Advantages / Disadvantages	
		Indexing	Query Processing
Cheng ke et.al [29]	Graph query tree with low cost index	Fast index creation Less space index	Faster query response
Li hadjlef et.al [32]	Authenticated tree based index structures	Index is updated dynamically	Low cost query execution Accurate query results
Li Feng et.al [15]	Steiner tree for keyword search in RDBMS	Less space index Less cost	Faster query response Accurate query results
Wei Hsu et.al [20]	R+ -tree in spatial data indexing (CDM)	Index takes more space	Query response time depends on query size and data size
Wa Cong et.al [45]	R-tree for spatial web object retrieval	Index takes more space	Query response depends on buffer size
Sandu Zeitouni et.al[55]	B+ -tree with indexing in network trajectory flows	For large networks index requires more space	Faster query response even query size and data size is big
Yuan Mitra et.al [49]	Graph lattice based indexing	Faster index construction Faster index update	Faster query results for sub graph querying
Maier Rattigan et.al[52]	Indexing network structures with shortest path trees	Index takes linear space	Accurate query results
Li Yi et.al []17]	B-tree based indexing for ranking queries on temporal data	Faster index construction Less index space	Faster query response Less cost
HSU Lee et.al [56]	K-tree based indexing to process reverse k-nearest neighbors(RKNN) queries	Index takes less space Less cost	Faster query response Accurate query results

Table 1 Graph based indexing techniques survey

graph querying and is compatible with any feature set. It is also used in false graph filtering, fast index lookups, quick index construction and maintenance, constructed using any choice of feature set which results in scalable and rapid sub graph querying infrastructure.

A network structured [50] index with graph clustering [54] is proposed which improves the scalability in processing the queries. K-medoids algorithm is applied which is simple and it is a discrete version of K-means data clustering method [51] and Girvan-Newman algorithm[52] is a clustering method based on centrality between edges. The index comprises of a series of node annotations along with a distance measure. Table 1 represents a detailed survey on graph based indexing techniques.

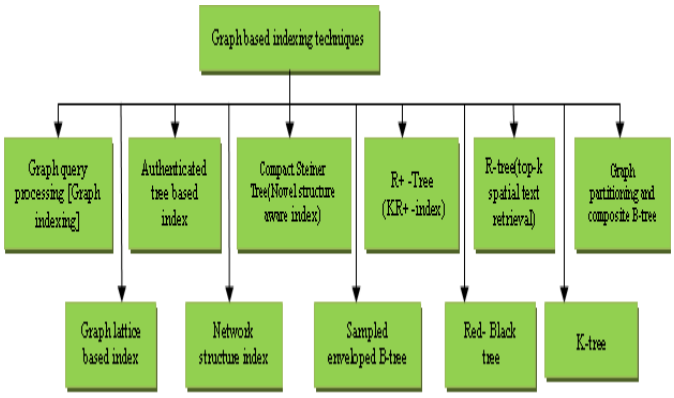


Figure 4 Graph based Indexing Techniques

IV CONCLUSION

To be concluded, the graph database based indexing results good than other traditional methods due to their improved performance on query processing with graph structures and also helps in reducing the indexing overhead because of using the commonality identification among the stored graph structured data. Furthermore, the graph structures allow faster index creation with less index space, less index creation time and faster index updating. Consequently, graph databases produce faster query response with less query execution cost and accurate query results.

REFERENCES

1. Richard K. Lomotey, Ralph Deters, Unstructured data mining: use case for Couch DB, Int. J. Big Data Intelligence. 2015.
2. D. Chen et al., “Real-time or near real-time persisting daily healthcare data into HDFS and elastic search index inside a big data platform,” IEEE Trans. Ind. Informatics, 2017, 13, pp. 595–606.
3. Gandomi and M. Haider, “International Journal of Information Management Beyond the hype : Big data concepts, methods, and analytics,” Int. J. Inf. Manage., 2015, 35, pp. 137–144.
4. Depeursinge, S. Duc, I. Eggel, and H. Müller, “Mobile medical visual information retrieval,” IEEE Trans. Inf. Technol. Biomed., 2012,16 pp. 53–61.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

5. Amer-Yahia S, Doan A, Kleinberg J, Koudas N, Franklin M Crowds, clouds and algorithms: exploring the human side of "big data" applications. Paper presented at the proceedings of the ACMSIGMOD international conference on management of data, Indianapolis, Indiana, USA,2010
6. Liu W, Peng S, Du W, Wang W, Zeng GS Security-aware intermediate data placement strategy in scientific cloud workflows. Know Inf Syst 2014,41,pp1–25
7. L. Ohno-Machado et al., "Finding useful data across multiple biomedical data repositories using Data Med," Nat. Genet., 2017,49, pp. 816–819.
8. Shan Liu, Lin Wang, Influence of managerial control on performance in medica linformation system projects: The moderating role of organizational environment and team risks, International Journal of project management, 2015.
9. Suthaharan, S.: Big Data Analytics. In: Machine Learning Models and Algorithms for Big Data Classification, vol.36. Integrated Series in Information Systems, .Springer US, 2016,pp. 31-75
10. Lomotey, Richard K., and Ralph Deters.: Unstructured data mining: use case for Couch DB. International Journal of Big Data Intelligence 2015, 2, pp168-182.
    Yu, Shanshan, Jindian Su, Pengfei Li, and Hao Wang."Towards High Performance Text Mining: A TextRank-
11. Yu, Kun-Ming, Sheng-Hui Liu, Li-Wei Zhou, and Shu-Hao Wu. "Apriori-based High Efficiency Load BalancingParallel Data Mining Algorithms on Multi-core Architectures." International Journal of Grid and High PerformanceComputing (IJGHPC), 2015, 7, pp77-99.
12. A. Siddiqa, A. Karim, and V. Chang, "SmallClient for big data: an indexing framework towards fast data retrieval," Cluster Comput., 2017, 20.
13. Zhu X, Huang Z, Cheng H, Cui J, Shen HT Sparse hashing for fast multimedia search. ACM Trans Inf Syst 2013, 31, pp1–24.
14. G, Feng J, Zhou X, Wang J providing built-in keyword search capabilities in RDBMS. VLDBJ 2011,20,pp1–19
15. Graefe G A survey of B-tree locking techniques. ACM Trans Database Syst 2010, 35.
16. Li F, Yi K, Le W Top-k queries on temporal data. VLDB J, 2010,19pp.715–733
17. Sandu Popa I, Zeitouni K, Oria V, Barth D, Vial S Indexing in-network trajectory flows. VLDB J,2011,20,pp643–669
18. Patrick .M, Katherine .P, Erin .F, Sutanay .C, A Chronological Edge-Driven Approach to Temporal Subgraph Isomorphis, arXiv [cs.DS] , 2018, pp 1-9.
19. Wei L-Y,HsuY-T, PengW-C, LeeW-C Indexing spatial data in cloud data managements. Pervasive Mobile Comput 2013,pp1–14.
20. Wu K, Shoshani A, Stockinger K Analyses of multi-level and multi-component compressed bitmap indexes. ACM Trans Database Syst 2010, 35, pp 1–52.
21. Mac Nicol R, French B Sybase IQ multiplex-designed for analytics. Paper presented at the proceedings of the thirteenth international conference on very large data bases, Toronto, Canada,2004,30.
22. Shang L, Yang L, Wang F, Chan K-P, Hua X-S Real-time large scale near-duplicate web video retrieval. In: Proceedings of the international conference on multimedia, 2010 pp 531–540
23. Wang Y, on contemporary denotational mathematics for computational intelligence. In: Gavrilova ML, Kenneth Tan CJ, Wang Y, Yao Y,Wang G (eds) Transactions on computational science II. Springer,Berlin,2008, pp 6–29
24. Rodríguez-García MÁ, Valencia-García R, García-Sánchez F, Samper-Zapater JJ Creating asemantically-enhanced cloud services environment through ontology evolution. Future Gener Comput Syst 2013, 32, pp295–306.
25. Bordogna G, Pagani M, Pasi G A dynamic hierarchical fuzzy clustering algorithm for information filtering. In: Herrera-Viedma E, Pasi G, Crestani F (eds) Soft computing in web information retrieval. Springer, Berlin, 2006,pp 3–23
26. Dittrich J, Blunschi L, VazSallesM MOVIES: indexing moving objects by shooting index images. Geoinformatica 2011, 15, pp 727–767.
27. Wai-Tat F Collaborative indexing and knowledge exploration: a social learning model. IEEE IntellSyst,2012, 27,pp 39–46
28. Cheng J, Ke Y, Fu AW-C, Yu JX (2011) Fast graph query processing with a low-cost index. VLDB J2011,20,pp521–539
29. Pokorný, J., Valenta, M. and Troup, M., Indexing Patterns in Graph Databases., Science and Technology Publications, 2018, pp 313-321
30. Williams, D.W., Huan, J., Wang, W.: Graph database indexing using structured graph decomposition. In: ICDE, 2007, pp. 976–985.
31. Li F, Hadjieleftheriou M, Kollios G, Reyzin L Authenticated index structures for aggregation queries. ACM Trans Inf Syst Secur 2010,13, pp 1–35.
32. Feng, J., Li, G., Wang, J., Zhou, L.: Finding and ranking compact connected trees for effective keyword proximity search in xml documents. Inform. Syst. ,2009
33. Hua, M., Pei, J., Fu A., W.-C., Lin, X., Leung, H.-F.: Top-k typicality queries and efficient query answering methods on large databases.VLDB J. 2009,18,pp809–835.
34. Kimelfeld, B., Sagiv, Y.: Finding and approximating top-k answers in keyword proximity search. In: PODS, 2006,pp. 173–182
35. Garg, N., Konjevod, G., Ravi, R.: A poly log arithmic approximation algorithm for the group steiner tree problem. J. Algorithms 2000, 37, pp 66–84
36. Adib. H, Ibrahim .A, Suhaidi .H, A survey on big data indexing strategies, research gate.net, 2015, pp 13-18
37. J. Wang, S. Wu, H. Gao, J. Li, B.C. Ooi, Indexing multi-dimensional data in a cloud system, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 2010, pp. 591–602.
38. Bin .F, Qing .Z, Xiao .F, An Efficient Graph-Based Spatio-Temporal Indexing Method for Task-Oriented Multi-Modal SceneData Organization, Geo-Inf, 2018, pp 1-18
39. T. Sellis, N. Roussopoulos, C. Faloutsos, The R+-tree: a dynamic index for multi-dimensional objects, in: Proceedings of the 13th International Conference on Very Large Data Bases, 1987, pp. 507–518.
40. N. Beckmann, H.P. Kriegel, R. Schneider, B. Seeger, The R*-tree: an efficient and robust access method for points and rectangles, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 1990, pp. 322–331.
41. Guttman, R-trees: a dynamic index structure for spatial searching, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 1984, pp. 47–57
42. Y.-T. Hsu, Y.-C. Pan, L.-Y. Wei, W.-C. Peng, W.-C. Lee, Key formulation schemes for spatial index in cloud data managements, in: Proceedings of the13th IEEE International Conference on Mobile Data Management, 2012, pp. 21–26.
43. Mohiuddin .A, Shiwen .C, Vagelis .H, A Comparative Study of Secondary Indexing Techniques in LSM-based No SQL Databases, 2018, pp 10-15
44. Wu D, Cong G, Jensen CS A framework for efficient spatial web object retrieval. VLDB ,2012,21,pp 97–822
45. De Felipe, I., Hristidis, V., Rishe, N.: Keyword search on spatial databases. In: ICDE, 2008,pp. 656–665
46. Cong, G., Jensen, C.S.,Wu, D.: Efficient retrieval of the top-k most relevant spatial web objects. PVLDB 2(1), 337–348(2009)
47. Guttman, A.: R-trees: a dynamic index structure for spatial searching. In: SIGMO, 1984,pp. 47–57
48. Yuan D, Mitra P Lindex: a lattice-based index for graph databases.VLDB J 2013,22,pp 229–252.
49. Trinajstic, N.: Chemical Graph Theory. Vol. 1, 2, 2nd edn. CRC Press, Boca Raton ,1992
50. Maier M, Rattigan M, Jensen D Indexing network structure with shortest-path trees. ACM Trans Knowledge Discovery Data 2011,5,pp 1-5
51. King, A. Graph Clustering with Restricted Neighbourhood Search. Master's thesis, University of Toronto,2004
52. Miao, Viliam, John, Patrick, High volumes of event stream indexing and efficient multi-keyword searching for cloud monitoring, Future Generation Computer Systems,2013,29,pp 1943–1962.
53. Matthew, Maier, David, Graph Clustering with Network Structure Indices, 2001.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

54. Sandu Popa I, Zeitouni K, Oria V, Barth D, Vial S Indexing in-network trajectory flows. VLDB J,2011,20,pp 643–66
55. Yeh S-C, Su M-Y, Chen H-H, Lin C-Y An efficient and secure approach for a cloud collaborative editing. J Netw Comput Appl 2013, 36, pp 1632–1641.
56. BalaAnand, M., Karthikeyan, N. & Karthik, S." Designing a Framework for Communal Software: Based on the Assessment Using Relation Modelling", Int J Parallel Prog (2018). https://doi.org/10.1007/s10766-018-0598-2
57. M.BalaAnand, S.Sankari, R.Sowmipriya, S.Sivaranjani "Identifying Fake User's in Social Networks Using Non Verbal Behavior", International Journal of Technology and Engineering System (IJTES), Vol.7(2), pg:157-161.
58. Maram, B., Gnanasekar, J.M., Manogaran, G. et al. SOCA (2018). https://doi.org/10.1007/s11761-018-0249-x
59. M. BalaAnand, N. Karthikeyan, S. Karthick and C. B. Sivaparthipan, "Demonetization: a Visual Exploration and Pattern Identification of People Opinion on Tweets," 2018 International Conference on Soft-computing and Network Security (ICSNS), Coimbatore, India, 2018, pp. 1-7. doi: 10.1109/ICSNS.2018.8573616
60. K. Anupriya, R. Gayathri, M. Balaanand and C. B. Sivaparthipan, "Eshopping Scam Identification using Machine Learning," 2018 International Conference on Soft-computing and Network Security (ICSNS), Coimbatore, India, 2018, pp. 1-7. doi: 10.1109/ICSNS.2018.8573687.
61. CB Sivaparthipan, N Karthikeyan, S Karthik "Designing statistical assessment healthcare information system for diabetics analysis using big data" Multimedia Tools and Applications, 2018
62. Zemedkun Solomon, C.B. Sivaparthipan, P. Punitha, M. BalaAnand, N. Karthikeyan "Certain Investigation on Power Preservation in Sensor Networks" ," 2018 International Conference on Soft-computing and Network Security (ICSNS), Coimbatore, India, 2018, doi: 10.1109/ICSNS.2018.8573688
63. M BalaAnand, N Karthikeyan, S Karthik, C.B. Sivaparthipan "A survey on BigData with various V's on comparison of apache hadoop and apache spark" - Advances in Natural and Applied Sciences, 2017