IFERP connecting engineers... developing research

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 3, Issue 4, April 2016

New Approach for Parallel Graph Computation Using Partition Aware Engine

[1] Prashant Patne [2] Vivek Takawale [3] Akash Tilak [4]Chandan Tiwari
[1][2][3][4] Department of Computer Engineering,RSCOE Pune.
Savitribai Phule Pune University
[1]-prashantpatne31@gmail.com [2] takawalevivek94@gmail.com [3] tilakakash7@gmail.com,
[4]tiwarichandan442@gmail.com

***Abstract:* - Graph partition quality affects the whole working of parallel graph computation systems. The quality of a graph partition is calculated by the equity factor and edge cut ratio. A balanced graph partition with small edge cut ratio is generally preferred since it reduces the costly network conversation cost. However, according to an empirical study on Graph, the achievement over well partitioned graph might be even two times worse than simple random partitions. This is because these systems only enhance for the easy partition strategies and cannot efficiently handle the increasing workload of local message transmission when a high quality graph partition is used. In this paper, we propose a novel partition aware graph estimation engine named PAGE, which qualify a new message processor and a dynamic concurrency control model. The new message processor simultaneously processes local and remote messages in a unified way. The dynamic model adaptively adjusts the compatibility of the processor based on the online statistics. The experimental evaluation demonstrates the prestige of PAGE over the graph partitions with various qualities.**

***Keywords*: Graph computation, graph partition, message processing**

I. INTRODUCTION

Massive big graphs are prevalent nowadays. Prominent examples include web graphs, social networks and other interactive networks in bioinformatics. The up to date web graph contains billions of nodes and trillions of edges. Graph structure can represent various relationships between objects, and better models complex data scenarios. The graph-based processing can facilitate lots of important applications, such as linkage analysis, community discovery, pattern matching and machine learning factorization models. With these stunning growths of a variety of large graphs and diverse applications, parallel processing becomes the defactorized graph computing paradigm for current large scale graph analysis. A lot of parallel graph computation systems have been introduced, e.g., Pregel, Giraph, GPS and Graph-Lab. These systems follow the vertex centric programming model. The graph algorithms in them are split into several super steps by synchronization barriers. In a super step, each active vertex simultaneously updates its status based on the neighbors' messages from previous super step, and then sends the new status as a message to its neighbors. With the limited workers (computing nodes) in practice, a worker usually stores a sub graph, not a vertex, at local, and sequentially executes the local active vertices. The computations of these workers are in parallel. However, in practice, a good balanced graph partition even leads to a decrease of the overall performance in existing systems As an example, when the edge cut ratio is about 3.48 percent in METIS, the performance is about two times worse than that in simple random partition scheme where edge cut is 98.52 percent. It indicates that the parallel graph system may not benefit from the high quality graph partition.

II. RELATED WORKS

Parallel graph computation is a popular technique to process and analyze large scale graphs. Different from traditional big data analysis frameworks (e.g., MapReduce), most of graph computation systems store graph data in memory and cooperate with other computing nodes via message passing interface. Besides, these systems adopt the vertex- centric programming model and release users from the tedious communication protocol. Such systems can also provide fault-tolerant and high scalability compared to the traditional graph processing libraries, such as Parallel BGL and CGMgraph. There exist several excellent systems, like Pregel, Giraph, GPS, Trinity. Since messages are the key intermediate results in graph computation systems, all systems apply optimization techniques for the message processing. Pregel and Giraph handle local message in computation component but concurrently process remote messages.

This is only optimized for the simple random partition, and cannot efficiently use the well partitioned graph. Based on Pregel and Giraph, GPS applies several other optimizations for the performance improvement. One for message processing is that GPS uses a centralized message buffer in a worker to decrease the times of synchronization. This optimization enables GPS to utilize high quality graph partition. But it is still very preliminary and cannot extend to a variety of graph computation systems. Trinity optimizes the global message distribution with bipartite graph partition techniques to reduce the memory usage, but it does not discuss the message processing of a single computing node. In this paper, PAGE focuses on the efficiency of a worker processing messages.

III. PROPOSED WORK

In our parallel graph computation technique whatever data file that can be provided as input that can be distributed into small blocks then Heuristic search, Edge Ranking, Edge cut strategies is applied on the data. After that processing data can be encrypted by using RSA algorithm for security purpose.

Encrypted data file is now stored on the server and that file can be acess by only authorized by registered user on the system. While achieving that file the data become decrypted and user can able to read that file. Whenever any unauthorized user try to acess the data then acess can be denied. In this way we can divide the data into relevant and un-relevant data file.

A. System Architecture

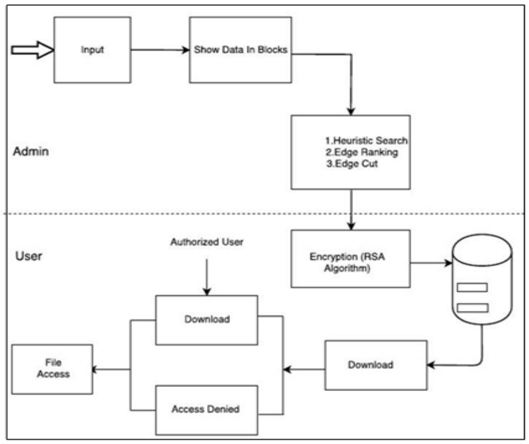

Fig 1. System Architecture

Fig. 1 presents the architecture of the system. The graph partition aware engine is a software module that takes a input data file from user, and returns a data file which can be classified as relevant data and un-relevant data as output.

- In this architecture the main components are user, admin and database server.
- Both Admin and User should have project software installed in his/her machine.
- At the start User uploads his data.
- Now execution process start where Admin divide the input data (which is given by user) into small blocks and applying the algorithms.
- All algorithms should apply successfully and discover the relevant data file.
- That file now stored on the database server in the encrypted format now admin task is over.
- In this stage only registered user is able to achieve the file which is stored on the database server.
- The file which user want that can automatically
- Decrypt at user side and user can download it.
- This Encryption and Decryption strategy can be used for security purpose.
- Now only authorized user is able to download the data if unauthorized user try to acess data th acess will be denied.

B. Heuristic Search Algorithm

Heuristic search is an AI search technique that employs heuristic for its moves. Heuristic is a rule of thumb that probably leads to a solution. Heuristics play a major role in search strategies because of exponential nature of the most problems. Heuristics help to reduce the number of alternatives from an exponential number to a polynomial number.

After the text rank the words are indexed with a particular score. A specific threshold value is provided to the system which is used for filtering of words from the result set. The top five words are shown to the user. For providing better user satisfaction we are going to provide a link where user can find the proper meaning of the result word from the forward dictionary, making it user friendly and efficient.

Description:

- Generate a possible solution which can either be a point in the problem space or a path from the initial state.
- Test to see if this possible solution is a real solution by comparing the state reached with the set of goal states.
- If it is a real solution, return. Otherwise repeat above steps.
- Might not always find the best solution but is there is no guarantee to find a good solution in reasonable in time.

C. RSA Algorithm

RSA is a widely used cryptosystem in the world. It is a public key cryptosystem which uses two kinds of key, private key and public key. Every user has both of the keys, a private one and a public one. If user A wants to send a message to B, he need B's public key to encrypt the message. After encrypted, the message is received by B, then B uses his private key to decrypt the message.

Description:

Encryption :
Sender A does the following:-

1. Obtains the recipient B's public key (n, e).
2. Represents the plaintext message as a positive integer m, 1 < m < n .
3. Computes the cipher text c = me mod n.
4. Sends the cipher text c to B.

Decryption :
Recipient B does the following:-

1. Uses his private key (n, d) to compute m = cd mod n.
2. Extracts the plaintext from the message representative m.

IV. MATHEMATICAL MODEL

Let S be a system that describes Lexical-LDA

S={…..}

Identify input as I
S={I,..}
Let I ={i1,i2,i3,…..id}

The input Text and user details.
Identify output as O
S = {I,O,…}

O= The receiver will partitioned data when he is authenticated.
Identify the processes as P

S={I,O,P,…..}

P={E,D}

E={parameter, id, input Text }
D={parameter, Skid, CTi} Identify the initial condition as Ic S={I, O, P, F,s ,Ic,…}
Ic=user should always be online and authorized.

Let Fs be the Functions used in the Proposed System where Fs={F1,F2,F3,F4,F5,F6}

Let F1 be a rule of S into W such that user logs into the system.

F1(s0)-->{u1,u2,...,un}€W

Let F2 be a rule of U into W such that user inputs a text.

F2(u0,u1,u2…un)--> (g0,g1,g2…gn) €W

Let F3 be a rule of E into D such that system receives the input value.

F3(g0,g1,g2…gn)-->{X,Y,Z} €D

Let F4 be the rule of A into D such that system process the input values using PAGE.

F4=(G)-->{Rv} €D

Let F5 be a rule of U into a such that comments is stored into database.

F5(g0,g1,g2…gn)-->{d0| Φd}€A

V. CONCLUSION

In this paper, we have classified the present data based on the previous data and also based on their on relevancy by using partition aware engine. In the adjusting model, we elaborated two heuristic rules to effectively extract the system characters and generate proper parameters. By using different algorithms the processing of parallel large graphs partition can be achieved successfully which can be very efficient.

REFERENCES

[1] Parallel multilevel graph partitioning, conference paper MAY 1996

[2] N. Backman, R. Fonseca, and U. C¸ etintemel, "Managing parallel- ism for stream processing in the cloud," in Proc. 1st Int. Workshop Hot Topics Cloud Data Process., 2012.

[3] P. Boldi and S. Vigna, "The webgraph framework I: Compression techniques," in Proc. 13th Int. Conf. World Wide Web, 2004.