International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 3 Issue: 2 484 - 488

Distributed Graph Isomorphism using Quantum Walks

Jayaprabha Yadav
Department of Computer Engineering
PES Modern College of Engineering
Pune, India
jayamailbox@gmail.com

Prof. Mrs. Aparna Junnarkar
Department of Computer Engineering
PES Modern College of Engineering
Pune, India
aparna.junnarkar@gmail.com

***Abstract*—**Graph isomorphism being an NP problem, most of the systems that solves the graph isomorphism are constrained with some classes of the graph, and do not work for all types of graphs in polynomial time. We exploited the two particle quantum walks on different classes of graphs including strongly regular graphs which are co-spectral in nature. We simulated two particle quantum walks on graph using distributed algorithm.

To show the effectiveness of the technique, we applied it to the large graphs derived from images using Delauney triangulation. The results show a remarkable speedup for large data. The two-particle quantum walks is implemented in map-reduce programming technique which scales the computation as the cluster get scaled to account Big data. We checked the isomorphism of the graphs with upto 100 vertices in polynomial time. The system is scalable to accept big inputs from any other domain in graph format**.**

Keywords- *Graph isomorphism; Distributed computing; Map-Raduce; Quantum Walks.*

*****

I. INTRODUCTION

In every application always there is a need to search and hence obviously we need to compare the objects. Nowadays the objects and the relations among them are getting more and more complex. This complexity can be represented by a data structure called graph. Two graphs are isomorphic if one can be obtained from another by relabeling of the vertices. Graph isomorphism is a basic methodology used in various applications for comparing objects.

The graph isomorphism is basically a hidden subgroup problem of the permutation group and is considered to be NP-complete. Checking weather two graphs are isomorphic is an NP-intermediate. A number of polynomial time methods have been suggested for finding isomorphic graphs. Some of these methods uses spectral analysis of graphs [1] and applies to specific classes of graph and doesn't work for co-spectral graphs which are strongly regular graphs. The best general classical algorithm to date runs in $O(C^{N\ 1/2\log N})$, C is a constant value and N is the number of vertices in the graphs being compared [2].

The two-particle quantum walks with hardcore bosons [3] can find isomorphism between two strongly regular graphs. The two particle QW is applied to graphs up-to 60 vertices and checked to be working correctly to check isomorphic graphs. The problem is to apply this method to graphs with more number of vertices.

We exploited a map-reduce algorithm for two particle quantum walks to calculate the evolution factor which can differentiate even the SRGs. This enables us to determine the isomorphism with highest responses in as O(nlogn*s*(1/p)) where n is m*m and m is no. of vertices in graph, s is the number of CPUs and p is the communication delay for one heartbeat time independent of the number of vertices in the graph. The technique is applicable to a variety of graphs that can be prepared from structures in image processing as well as other domains. Our approach is not restricted to any particular method or dataset; rather it provides the basis for scaling the number of traditionally compute-intensive graph processing operators from the hundreds or at most thousands applied in current practice to millions. We first applied the Delaunay triangulation method on images using batik library to derive graphs from images. Then we derived the adjacency matrix of the image. We demonstrate the efficacy of the approach by scaling graph isomorphism to multiple graph classes employing Delaunay triangulation preprocessing of the images representing graphs. The isomorphic images can be detected even if the target image graph undergoes multiple transformations.

We believe the present work will accelerate the state of the art in object detection by increasing the number of visual categories by an order of magnitude or more while simultaneously reducing run times by a comparable factor. We demonstrate our approach in an implementation that achieves respectable performance on a standard benchmark for graph isomorphism, and exhibits graceful progress in performance with larger, automatically generated datasets consisting of tens of thousands of graph data.

II. RELATED WORK

These include probabilistic methods [4] and graph-spectral methods which utilizes the Eigen values and eigenvectors of the Laplacian matrix [5]. The auxiliary graph

method by David Emms [6] has the complexity dominated by the simulation of the walk and is $O(|V|^6)$.By comparison, for the Umeyama algorithm [7] the complexity of computing the spectra of the two adjacency matrices is $O(|V|^3)$and the complexity of using the Hungarian search method is again $O(|V|^3)$. This is typical of most graph-spectral methods. Other approximate algorithms have similar complexity. For instance, Gold and Rangarajan's [8] algorithm has complexity $O(|E_G| \times |E_H|)$ where E_G and E_H are edges in G and H graph respectively. There are of course more sophisticated inexact graph matching algorithms available. Douglas and Wang [9] have recently explored the use of discrete quantum walks for graph isomorphism. Their idea is to use the probability amplitudes associated with the states of walks on separate graphs as node-attributes. The amplitudes for corresponding steps of the walk on different graphs are compared to establish isomorphism or similarity. The algorithm has complexity $O(|V|^7)$.

Two-particle interacting boson walks distinguish all non-isomorphic pairs of SRGs, the SRGs with up to 64 vertices [3]. If the two-particle quantum walks can check for all types of isomorphic graphs, then the Graph isomorphism is a polynomial problem and not an NP can be proved.

III. TECHNICAL DETAILS

The architecture described in this paper applies to a wide range of graph types, e.g., spectral and Strongly Regular graphs (SRG). The application and experiments presented here make use of the map-reduce model (MR) of Hadoop. Many other graph matching applications can be adapted to use our approach, including hidden sub graph detection [10], hidden group in financial transaction network [11], and communication networks [12].

A. One particle continuous time quantum walks

The state space for the continuous-time quantum walk on a graph, G = (V, E), is the set of vertices, V, as is the case for the classical random walk. In addition, transitions only occur between adjacent vertices. If the walk is at a vertex u, it moves to adjacent vertices at a rate proportional to 1/d(u), where d(u) is degree at u. The basis states for the continuous-time quantum walk are vectors corresponding to particular vertices, as is the case for the classical random walk and unlike the discrete-time quantum walk where basis states correspond to edges. The basis state corresponding to the walk being at $u \in V$ is written, in Dirac notation, as $|u\rangle$. A general state of the walk is a complex–linear combination of these basis states and so the state of the walk at time t is given by a vector, which we write component wise as

$$|\psi_t\rangle = \sum_{u \in V} \alpha_u(t)|u\rangle \qquad (1)$$

Unlike the classical walk, the quantum walk is not a Markov chain. Given an initial state for the walk, $|\psi_0\rangle$, Eq. (1) can be solved to give

$$|\psi_t\rangle = e^{-iLt}|\psi_0\rangle \qquad (2)$$

,where L=D−A is the Laplacian matrix, A is the adjacency matrix and D is the diagonal degree matrix.

If we restrict ourselves to single-particle states, we find that adjacency matrix elements give Hamiltonian,

$$\langle i \mid H \mid j \rangle = -A_{ij} \qquad (3)$$

Hence, we can easily identify the a single-particle Hamiltonian

$$H_{1P} = -A \qquad (4)$$

And Quantum Walk time evolution operator can be defined as

$$U = e^{-itH} \qquad (5)$$

An SRG is a graph in which (a) all vertices have the same degree, (b) each pair of neighboring vertices has the same number of shared neighbors, and (c) each pair of non-neighboring vertices has the same number of shared neighbors. This definition permits SRGs to be categorized into families by four integers (N, k, λ, μ), each of which might contain many non-isomorphic members. Here, N is the number of vertices in each graph, k is the degree of each vertex (k regularity), λ is the number of common neighbors shared by each pair of adjacent vertices, and μ is the number common neighbors shared by each pair of nonadjacent vertices.

The adjacency matrix of all strongly regular graphs satisfies the useful relation [13]

$$A^2 = (k-\mu)I + \mu J + (\lambda - \mu)A \qquad (6)$$

where I is the identity matrix, A is adjacency matrix, and J is the matrix of all 1s. We can write

$$A_n = \alpha_n I + \beta_n J + \gamma_n A \qquad (7)$$

,where α, β, γ are functions only of the family parameters.

That is, all SRGs of the same family have the same coefficients and same Hamiltonian and hence the algorithm based on single-particle quantum evolution fails to distinguish any non-isomorphic SRGs that are in the same family.

B. Two particle continuous time quantum walks

First, we note that we may write the Hamiltonian for any two-boson Quantum Walk as

$$H_{2B} = -\frac{1}{2}(I+S)(A \oplus A) + UR \qquad (8)$$

Where $A \oplus A = (A \otimes I) + (I \otimes A)$ is a Kronecker sum, the matrix special case of a direct sum, and

$$S = \sum_{i,j} |ij\rangle\langle ji| \qquad R = \sum_{i} |ii\rangle\langle ii| \qquad (9)$$

Using equation (5) we get,

$$U = e^{-it\,[-2(I+S)(A \oplus A) + UR]} \qquad (10)$$

Expanding as a power series in t, we have

$$U = \sum_{n=0}^{\infty} \frac{(-it)^n}{n!} [-\frac{1}{2}(I+S)(A \oplus A) + UR]^n \quad (11)$$

Numerically, it is determined [3] that a sixth-order expansion was necessary, and that the term B^2RB^3 can be used to distinguish the graphs up to N=40 vertices, where $B = A \oplus A$. The component of evolution factor that differentiate the value of evolution factor can be used to represent the graph so that it can be compared with other graph's evolution factor. This solution is used to formulate the MapReduce algorithm to differentiate the graphs with number of vertices upto 100.

C. Map-reduce algorithm for Quantum Walks

MapReduce is a programming model invented for distributed data processing on large clusters. As part of two particle Quantum Walk on graph, two algorithms MapReduceKron and MapReduceMatrix are created for computing Kronecker sum B and multiplications of matrices. An algorithm for calculations on matrices requires a large amount of memory to store the input operands, intermediate results and output. Traditionally calculations on large arrays require a uniform address space on a single machine and these operations are performed on supercomputers with software customized to exploit its particular memory and interconnect architecture. When memory is exhausted during a computation number of packages may have a poor performance or may not work at all, as memory available in most machines is insufficient by orders of magnitude.

In this paper we present a distributed implementation of two-particle quantum walks for large graphs on commodity hardware using a shared compute layer. Our prototype implementation runs on Hadoop clusters. Hadoop is an open source distributed computing framework developed by Apache Software Foundation. It includes fault tolerant, distributed file system namely HDFS, designed for high-throughput access to very large data sets. It also includes an implementation of Map-Reduce, a programming model designed for processing big data sets on large clusters.

Input for the map reduce is <key, value> pair i. e. two files containing adjacency matrices derived from two different graph data. It is actually edge list of the graph.

<vertex1, vertex2, 1>

$$A_{i,j} = \{X_{i,j} | < i, j > is\, an\, edge\, in\, the\, graph\}$$

The double occupancy matrix R is calculated for every graph,

$$R_{i,j} = \{X_{ii,jj} \mid X_{i,j} \in A\ adjacency matrix\}$$

Kronecker sum B is calculated using map-reduce implementation.

$$B = \{x_{i,j} | x_{i,j} = \sum_{i,x,y}^{m} < i*m+x, i*m+y > < x*m+i, y*m+i > \}$$

A distributed multiplication using map-reduce programming method is implemented. Evolution factor can be calculated using the differentiating component as B2RB3.

The evolution factors for two graphs are U1and U2 such that they are elements of the matrices. These values are placed in the separately and sorted. The difference of these lists if zero then the two graphs are isomorphic else are non-isomorphic.

It is found that evolution factor can identify distinct graphs upto 64 vertices for SRGs and upto 100 vertices for images. T he results are discussed in next section.

IV. Experimental Results

In the following, we use the term one-particle algorithm to refer to the graph-isomorphism algorithm described in [2] utilizing graphs in SRG dataset. To execute map-reduce system we prepare the cluster of 5 machines having dual core and 2GB RAM, single rack based structure and Hadoop 2.6.0 distributed system setup. One of the machine is namenode and resource-manager while others slaves to work as datanode and nodemanager. Replication factor set to four. By performing several experiments, we compare the performance of the one-particle algorithm with the distributed two-particle algorithm described in the previous section, utilizing map-reduce model of programming as mentioned earlier. First, we demonstrate that the one-particle algorithm compares favorably with the two-particle algorithm on the SRG dataset. Second, we show that two-particle distributed algorithm can scale graph-isomorphism to hundreds of thousands of graph nodes and provide insight into the trade-offs involving accuracy, memory and computation time. We use our system for data derived from real-world data. For this we take images from the COIL database and construct Delaunay triangulations. We consider 12 different objects from the database. We used our system to compare images.

A. *Dataset and implementation Details*

We first employed the SRG dataset upto 64 vertices to test our implementation, which work correctly to identify the non-isomorphic SRGs from the same class of SRGs. The system required more time as we checked on a cluster of multiple nodes. In the case of a distributed system the network overhead is more than that of computations for small datasets. But the task of complex computation can be distributed in easy steps and same application can handle big data. We can see in the fig. 1 that the same number of vertices requires more time if the distributed file system block size reduced.

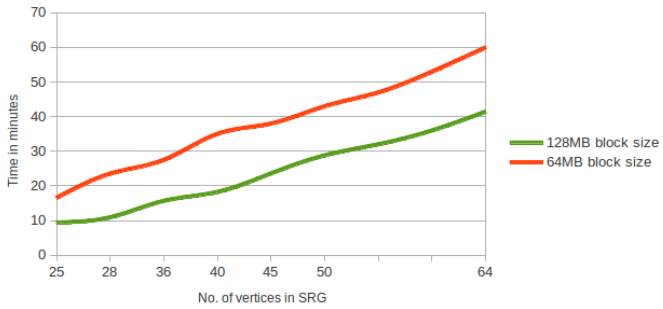

Figure 1. The time required for checking isomorphism to the number of vertices of graph with sparse connectivity and dense connectivity.

We employed the standard image dataset, COIL-20, to test our system. The COIL-20 dataset contains images from different categories with 20 images for training and validation. The triangulated graphs can be used to check isomorphism. We can consider the triangulated images close to SRG class of graph because certain set of triangular graphs are also SRGs.

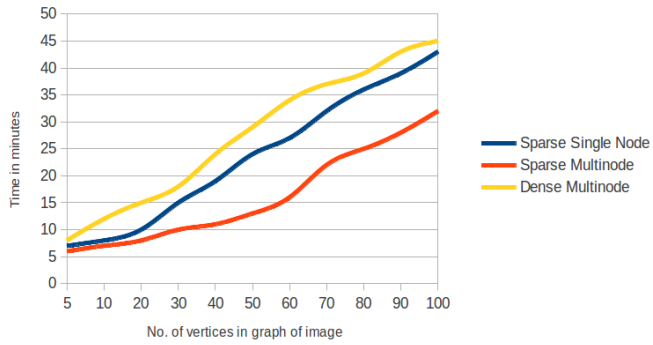

Figure 2. The time requires for detecting isomorphism of SRGs from E. Spence Dataset [14] with 64MB block size and 128MB block size.

Fig. 2 shows the results for sparse matrix that is the graph with less number of edges requires less time as compared to that for a dense graph, which has more number of edges. In dense graph the computations are more as we are considering the adjacency edges for the calculation of evolution factor. The Figure 2 shows the results for dense graphs derived from images on multiple node clusters.

The continuous time quantum walks are better than discrete time quantum walks as continuous time walks corresponds to vertices [2]. But while implemented in map-reduce it is observed that more dense the adjacency matrix, more time it takes for distributed computation. That is the time required is more as the connectivity of the graph is increased.

B. *Accuracy, speed and memory*

Our implementation is able to detect isomorphism between graphs of all classes with 100% accuracy. As the number of vertices is increased, the input is increased and thus number of map tasks is increased so the time required to collect result from multiple nodes is greater and hence total time required for computation is increased. But the system withstands big input data.

CONCLUSIONS

Our key contribution is a distributed approach to graph isomorphism that replaces computational density with distributed simple operations by using an efficient Map-Reduce design. This approach is applicable to a variety of graphs of different classes and derived from different application domain. Through extensive empirical tests on distributed graph isomorphism system, we have shown that (a) the system performs comparably better to the original single machine algorithm for big data, (b) performance upgrades gracefully as the number of vertices in graph is increased, and (c) up to 100 vertices graph can be processed on a single machine in polynomial time.

REFERENCES

[1] B. D. McKay, Practical graph isomorphism, Congr. Numer. 30 (1981) 45–87.

[2] A. Robles-Kelly, E. R. Hancock, Graph edit distance from spectral seriation, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 365–378.

[3] John King Gamble, Mark Friesen, Dong Zhou, Robert Joynt, and S. N. Coppersmith, “Two-particle quantum walks applied to the graph isomorphism problem", Phys. Rev. A 81, 052313 - Published 13 May 2010.

[4] W. J. Christmas, J. Kittler, M. Petrou, Structural matching in computer vision using probabilistic relaxation, IEEE Trans. Pattern Anal. Mach. Intell. 17 (8) (1995) 749–764.

[5] A. Robles-Kelly, E. R. Hancock, Graph edit distance from spectral seriation, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 365–378.

[6] D. Emms, S. Severini, R. C. Wilson, and E. R. Hancock, “Graph matching using the interference of continuous-time quantum walks", Pattern Recog. 42, 1988 (2009).

[7] S. Umeyama, An eigen decomposition approach to weighted graph matching problems, Pattern Anal. Mach. Intell. 10 (5) (1988) 695–703.

[8] S. Gold, A. Rangarajan, A graduated assignment algorithm for graph matching, IEEE Trans. Pattern Anal. Mach. Intell. 18 (4) (1996) 377–388.

[9] B. Douglas and J. Wang, Journal of Physics A: Mathematical and Theoretical 41, 075303 (2008).

[10] R. Jozsa, Quantum factoring, discrete logarithms, and the hidden subgroup problem, Comput. Sci. Eng. 3 (2) (2001) 34–43.

[11] Yuhua Li, Dongsheng Duan, Guanghao Hu, Zhengding Lu,"Discovering Hidden Group in Financial Transaction Network Using Hidden Markov Model and Genetic Algorithm", Fuzzy System and knowledge Discovery,2009,pp. 253-258.

[12] J. Baumes, M. Goldberg, M. Magdon-Ismail and A. Wallace, "Discovering Hidden groups in communication Networks", Intelligence and security Information, 2004.

[13] C. Gosil and G. Royle, Algebraic Graph Theory (Springer, New York, 2001)

[14] E. Spence, Strongly regular graphs on at most 64 vertices. Available at: http://www. maths. gla. ac. uk/~es/srgraphs. html].

[15] Columbia Object Image Library (COIL-20)," S. A. Nene, S. K. Nayar and H. Murase, Technical Report CUCS-005-96, February 1996.