2012 IACSIT Hong Kong Conferences
IPCSIT vol. 30 (2012) © (2012) IACSIT Press, Singapore

An Application of Tensor Product Expansion for Hierarchization of Graph Model

Hidetomo Suzuki [1+], Atushi Ohta [2] and Kokichi Tsuji [2]

[1] Graduate School of Information Science and Technology, Aichi Prefectural University
[2] Department of Information Science and Technology, Aichi Prefectural University

Abstract. Graph theory is useful tool for analyzing data in database to discover rule behind it, modeling topological aspect on a system and etc. Due to the enormous increase in database sizes and system scale, as well as graph size is becoming more and more vast. Modeling and analyzing graphs require a large memory and additional computation time. In this paper, modified tensor product expansion is applied to graph model for reducing graph size. Tensor product expansion is an algorithm to decompose a matrix into vectors. On the other hand, proposed tensor product expansion decomposes a matrix into smaller matrices. As a result, applying proposed algorithm to graph model makes it possible to analyze model's property of every layer's graph which size is smaller than original graph size separately because common parts in the graph are grouped into layers. The algorithm is with non-linear optimization in it. Therefore, computational time rely on the non-linear optimization algorithm and number of matrices.

Keywords: graph, tensor product expansion, non-linear optimization.

1. Introduction

Graph is graphical and mathematical tool. Graph models show structure of a system to us by image and it's possible to be analyzed by mathematics. Graph theory has been used for wide variety fields. For example data mining [1], system modelling [2], social network [3] and etc. With system, data in database or something is becoming bigger [4], graph model is becoming more and more vast. Large-scale graph model can't show the background rules simply. Extracting important information from large-scale graph efficiently must be possible for wide variety application. An answer of the problem is decomposing the model into smaller models.

There is an algorithm to decompose a space created by tensor product of vectors into vectors called tensor product expansion [5]. A tensor is a multidimensional array. More formally, an N -way or N th-order tensor is an element of the tensor product of N vector spaces, each of which has its own coordinate system. A first-order tensor is a vector, a second-order tensor is a matrix and tensors of order three or higher are called higher-order tensors. Tensor product expansion's computational procedure is to apply non-linear optimization iteratively to derive the expansion coefficients and the expansion vectors. Generally, the problems in non-linear optimization have been that the process may not converge or a solution other than the optimal may be derived, depending on the given initial value, or the computational complexity and the computation time become tremendous when the size of the data is enlarged.

Tensor product expansion can't be applied to graph model as is. In order to handle the problem, the authors devised a method of calculating the tensor product expansion to decompose a matrix into smaller matrices. Using the algorithm is possible to analyzing smaller graphs instead of treating original large-scale graph. However, problems about initial value and computational time also still exist for the new algorithm. They must be gotten fixed.

+ Corresponding author. Tel.: + 09096635816.
E-mail address: *zuqqhi2@gmail.com.*

2. Preliminary

In this section important concepts are shown for modified tensor product expansion with confirming notation. The concepts are pared to the essentials.

2.1. Definition of graph

In this part graph definition is shown. The following definition captures the situation where the direction of the edges is taken into account. First, simple graph is defined as follows.

Definition 1. ***Directed graphs***. A pair $G = (V, E)$ is an ordered pair of sets. Elements of V are called vertices, and elements of $E \subseteq V \times V$ are called edges. We refer to V as the vertex set of G, with E being the edge set. The cardinality of V is called the order of G, and $|E|$ is called the size of G. A directed edge is an edge such that one vertex incident with it is designated as the head vertex and the other incident vertex is designated as the tail vertex. A directed edge (u, v) is said to be directed from its tail u to its head v. A weight between $u, v \in V$ is described by $w(u, v) \in \Re$.

Representing a graph as a matrix is very efficient in some cases. The matrix's definition is as follows.

Definition 2. ***Adjacency matrix***. A graph G is a directed graph with vertices $V = \{v_1, \cdots, v_n\}$ and edge set E. The adjacency matrix of G is the $n \times n$ matrix $A = A^+ - A^- = [a_{ij}]$ defined by

$$a_{ij} = \begin{cases} w(v_i, v_j), & if \ \ (v_i, v_j) \in E. \\ -w(v_j, v_i), & if \ \ (v_j, v_i) \in E., \\ 0, & otherwise. \end{cases} \tag{1}$$

$$a_{ij}^+ = \begin{cases} w(v_i, v_j), & if \ \ (v_i, v_j) \in E. \\ 0, & otherwise. \end{cases}, a_{ij}^- = \begin{cases} w(v_j, v_i), & if \ \ (v_j, v_i) \in E. \\ 0, & otherwise. \end{cases}$$

The matrix $A^+ = [a_{ij}^+]$ is called forward adjacency matrix and the matrix $A^- = [a_{ij}^-]$ is called backward adjacency matrix.

2.2. Tensor product

Only most basic definition of tensor product is given below. Because properties of tensor product are very important with analyzing graph after applying proposed algorithm to graph, see more detail [6].

Definition 3. ***Tensor Product***. The tensor product of $A \in \Re^{m \times n}$ and $B \in \Re^{p \times q}$ is defined as the matrix

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix} \in \Re^{mp \times nq}. \tag{2}$$

3. Tensor Product Expansion for Graph Decomposition

In this section we propose a new algorithm to decompose a matrix which is represented by tensor product with matrices into them. The algorithm based on iterative updates of matrices using non-linear optimization.

3.1. Mathematical formulation

We first need to define cost functions that quantifies the quality of the approximation. Such a cost function can be represented by Euclidean distance between the original matrix and a matrix that is constructed by tensor product with some matrices,

$$J = \left(\hat{A} - \bigotimes_i A_i \right)^2 \tag{3}$$

This is lower bounded by zero, and clearly vanishes if and only if $\hat{A} = \otimes_i A_i$. Optimization is done for two times to decompose graph because we need to decompose forward and backward adjacency matrix

respectively. If a optimization problem with $A(=A^+ - A^-)$ is only solved, information of self-loop and hierarchy disappear. We now consider two formulations of graph decomposition as optimization problems:

$$\min_{A_i^+} J^+ = \left(\hat{A}^+ - \bigotimes_i A_i^+ \right)^2, \ \min_{A_i^-} J^- = \left(\hat{A}^- - \bigotimes_i A_i^- \right)^2 \tag{4}$$

3.2. Normalize result matrices

Note that problems results of eq.(4) have degree of freedom with multiplier factor. It's not always true that they are expected results. They depend on their initial values on iterative method. If target graph has some properties, we have potential to can obtain the solution corresponding to expected results to add conditions to the problems. If target graph doesn't have special properties, we need to apply equitable distribution of matrix norm defined as follows to results to obtain same solution irrespective of initial values,

$$A_k \leftarrow w_k A_k, \quad w_k = \left(\sqrt{\frac{\prod_{i=1}^{n[-k]} \left\| A_i \right\|_F}{\left\| A_k \right\|_F^{n-1}}} \right) \tag{5}$$

In this paper norm of matrix A is represented as follows

$$\left\| A \right\|_F = \sqrt{\sum_{i,j} \left(A_{ij} \right)^2} \tag{6}$$

3.3. Update rule

Although the function $(\hat{A} - \otimes_i A_i)^2$ is convex in A_i only, they are not convex in both variables together. Therefore it is unrealistic to expect an algorithm to solve problems in the sense of finding global minima. However, there are many techniques from numerical optimization that can be applied to find local minima. Any techniques can be applied, but in this paper we use relaxation method because the algorithm is easy to implement and avoid to obtain local minimum.

Eq.(4) is formulated as follows in terms of elements,

$$\min_{A_i^+} J^+ = \sum_{l_{11} l_{12} \cdots l_{n2}} \left(\hat{A}^+_{\pi(l_{11},\cdots,l_{n1}),\pi(l_{12},\cdots,l_{n2})} - \prod_i A^+_{i,l_{i1},l_{i2}} \right)^2 \tag{7}$$

Where, $\pi(\cdots)$ is a function which returns row or column number on a matrix using arguments. Much the same is true on backward adjacency matrix. Solving the cost function for $A^+_{i,l_{i1},l_{i2}}$ by setting partial differential with $A^+_{i,l_{i1},l_{i2}}$ equal to zero gives following update rule,

$$A^+_{i,l_{i1},l_{i2}} = \frac{\sum_{l_{11},\cdots,l_{n2}} A^+_{\pi(l_{11},\cdots,l_{n1}),\pi(l_{21},\cdots,l_{n2})} \prod_j^{[-i]} A^+_{j,l_{j1},l_{j2}}}{\sum_{l_{11},\cdots,l_{n2}} \prod_j^{[-i]} \left(A^+_{j,l_{j1},l_{j2}} \right)^2} \tag{8}$$

Where, $[-i]$ means that i th matrix is ignored. Other matrices' elements are as constant value on eq.(8).

Here we propose a algorithm to decompose given graph's adjacency matrix into adjacency matrices $\{A_1, \cdots, A_n\}$, that is, $A = A^+ - A^- = (\otimes_i A_i^+) - (\otimes_i A_i^-)$. The algorithm is to apply non-linear optimization iteratively to derive the expansion matrices decreasing cost function eq.(3). The detail of the algorithm is as follows using eq.(4) (5) (6) (8),

Algorithm 1. ***Tensor Product Expansion For Graph Decomposition***.

Input: Graph's adjacency matrix A^+ and A^- for decomposing, and margin error δ.

Output: Adjacency matrices $\{A_1^+, \cdots, A_n^+\}$ and $\{A_1^-, \cdots, A_n^-\}$.

Step 1. Initialize $A_1^+, \cdots, A_n^+$ and $A_1^-, \cdots, A_n^-$ randomly.

Step 2. Obtain A_i^+ using eq.(8) by other matrices are as constant.

Step 3. Apply Step 2 to $A_1^+, \cdots, A_n^+$.

Step 4. While Euclidean distance J^+ is greater than δ, repeat step 2 and 3.

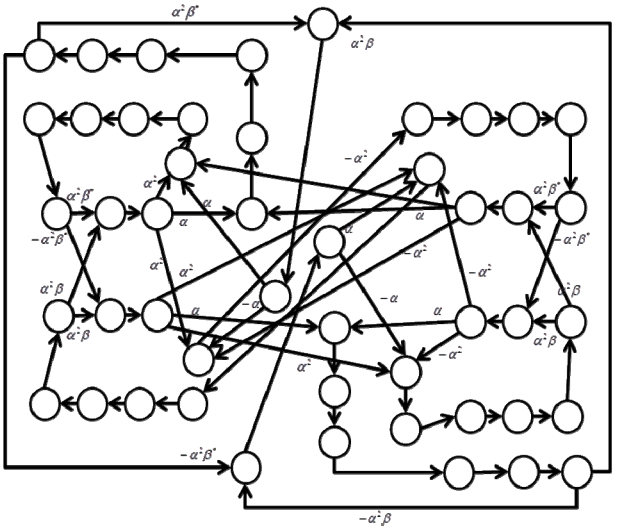
Step 5. Apply equitable distribution of matrix norm defined as eq.(5).

Step 6. Apply Step 1-5 to backward adjacency matrix $A_1^-, \cdots, A_n^-$.

This algorithm can stop and achieve a solution any time if given matrix can be decomposed. However, this algorithm can't guarantee obtaining signs of solution corresponding to them. Note that solution might be local minimum if other update rule is used. In that case we need use a technique to avoid local minimum at Step 2 and 3. An example of application of Algorithm1 is shown Fig.1 and Fig.2.

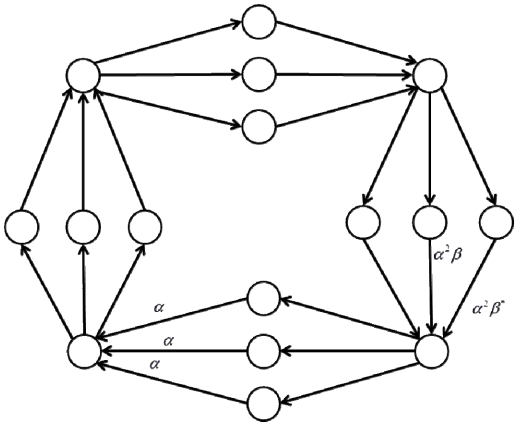
4. Results

Results that apply proposed algorithm to variety of target matrix sizes are shown Fig. 3. It is plotted with sec on the y-axis on logarithmic scale. It represents that computational time is becoming bigger exponentially for target matrix scale.

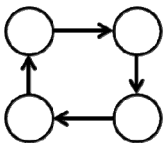


G

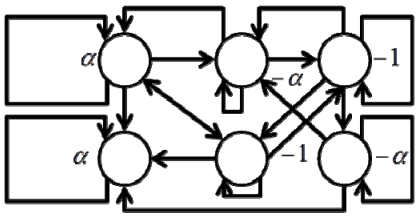
Fig. 1: Original Graph



G_1



G_2



G_3

Fig. 2: Decomposed Graphs

5. Conclusions

In this paper, we have proposed a tensor product expansion to decompose a graph into graphs. This work makes it possible to treat large-scale graph easily. However, proposed algorithm's computational time is becoming bigger exponentially for target graph scale. This means applying the algorithm to huge graph is difficult. Improving the algorithm's computational complexity must be done. Furthermore, there are no methods to know whether target graph can be decomposed on ahead. Then, for the future, researching relations between property of target matrix and the algorithm is needed in order to make the algorithm more efficient.

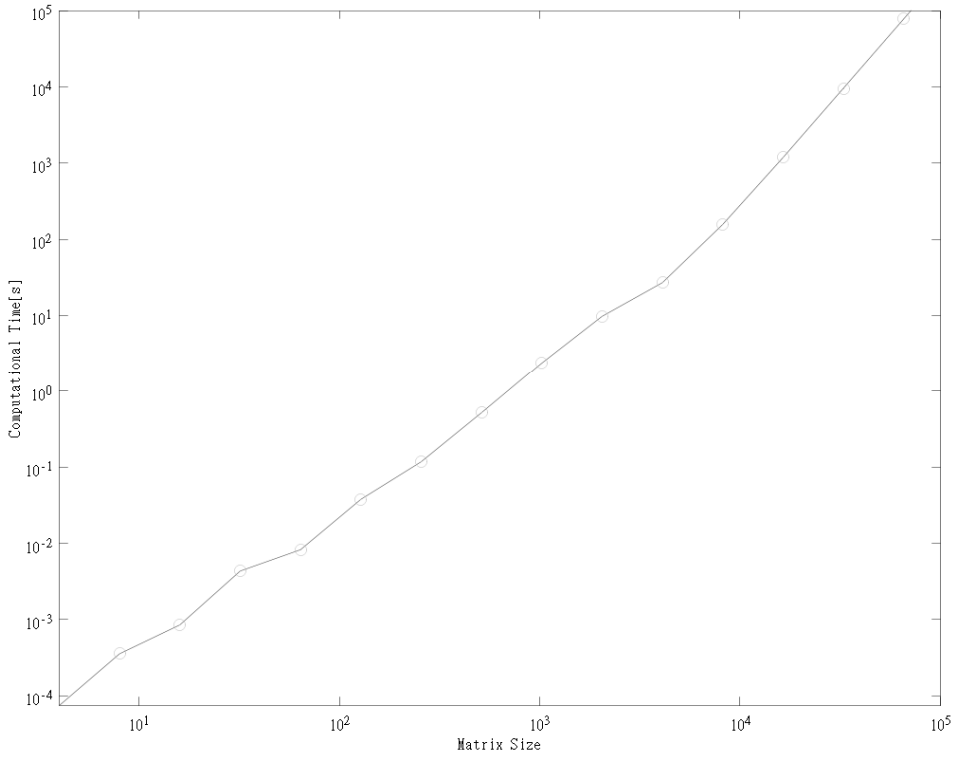


Fig. 3: Computational Time

6. References

[1] Peter A. Bath, Cheryl Craigs, Ravi Maheswaran, John Raymond, and Peter Willett. Use of Graph Theory to Identify Patterns of Deprivation and High Morbidity and Mortality in Public Health Data Sets. *J Am Med Inform Assoc.* 2005, **12** (6):630-641.

[2] Zan Huang, Wingyan Chung, and Hsinchun Chen. A Graph Model for E-Commerce Recommender Systems. *Journal of The American Society for Information Science and Technology*. 2004, **55** (3):259-274.

[3] M. E. J. Newman, D. J. Watts, and S. H. Strogatz. Random graph models of social networks. *Proceedings of the National Academy of Sciences of the United States of America*. 2002, **99** (Suppl 1):2566-2572.

[4] Bin Wu, Yuxiao Dong, Qing Ke, and Yanan Cai. A Parallel Computing Model for Large-Graph Mining with MapReduce. *Proceeding of 7th International Conference on Natural Computation*. 2011:43-47.

[5] Tamara G. Kolda, and Brett W. Bader. Tensor Decompositions and Applications. *SIAM Review*. 2009, **51**(3):455-500.

[6] Alan J. Laub. *Matrix Analysis for Scientists and Engineers*. Society for Industrial Mathematics. 2004.

[7] Olivier Duchenne, Francis Bach, Inso Kweon and Jean Ponce, A Tensor-Based Algorithm for High-order Graph Matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011, **33**:2383-2395.