ASIAN JOURNAL OF CIVIL ENGINEERING (BUILDING AND HOUSING) VOL. 7, NO. 2 (2006)
PAGES 125-137

A UNIFIED METHOD FOR EIGENDECOMPOSITION OF GRAPH PRODUCTS

A. Kaveh* and H. Rahami
Department of Civil Engineering, Iran University of Science and Technology,
Tehran-16, Iran

ABSTRACT

In this paper, a unified method is developed for calculating the eigenvalues of the weighted adjacency and Laplacian matrices of three different graph products. These products have many applications in computational mechanics, such as ordering, graph partitioning, and subdomaining of finite element models.

Keywords: adjacency, Laplacian, Cartesian product, strong Cartesian product, direct product, eigendecomposition, regular graph, graph product

1. INTRODUCTION

Graph theory has a long history, and its applications in structural mechanics and in particular nodal ordering and graph partitioning are well documented in the literature, Kaveh [1-2].

Algebraic graph theory can be considered as a branch of graph theory, where eigenvalues and eigenvectors of certain matrices are employed to deduce the principal properties of a graph. In fact eigenvalues are closely related to most of the invariants of a graph, linking one extremal property to another. These eigenvalues play a central role in our fundamental understanding of graphs. There are interesting books on algebraic graph theory such as Biggs [3], Cvetković et al. [4], and Godsil and Royle [5].

One of the major contributions in algebraic graph theory is due to Fiedler [6], where the properties of the second eigenvalue and eigenvector of the Laplacian of a graph have been introduced. This eigenvector, known as the *Fiedler vector* is used in graph nodal ordering and bipartition, Refs. [7-9].

General methods are available in the literature for calculating the eigenvalues of matrices, however, for matrices corresponding to special models, it is beneficial to make use of their extra properties.

In this paper, a unified approach is developed for calculating the eigenvalues of the adjacency and Laplacian matrices of three different graph products. These methods have

* Email-address of the corresponding author: alikaveh@iust.ac.ir

many applications in computational mechanics, such as ordering, graph partitioning, and subdomaining finite element models, Kaveh and Rahami [10,11].

2. DEFINITIONS

2.1 Definitions from Graph Theory

A *graph* G(N,E) consists of a set of elements, N(G), called *nodes* and a set of elements, E(G), called *edges*, together with a relation of incidence which associates two distinct nodes with each edge, known as its *ends*. Two nodes of a graph are called *adjacent* if these nodes are the end nodes of an edge. An edge is called *incident* with a node if it is an end node of the edge. The *degree* of a node is the number of edges incident with the node. A *subgraph* G_i of a graph G is a graph for which $N(G_i) \subseteq N(G)$ and $E(G_i) \subseteq E(G)$, and each edge of G_i has the same ends as in G. A *path* of G is a finite sequence $P_i = \{n_0, m_1, n_1, ..., m_p, n_p\}$ whose terms are alternately distinct nodes n_i and distinct members m_i of G for $1 \le i \le p$, and n_{i-1} and n_i are the two ends of m_i. A *cycle* is a path $(n_0, m_1, n_1,..., m_p, n_p)$ for which $n_0 = n_p$ and $p \ge 3$; i.e. a cycle is a closed path. A cycle graph with n nodes is denoted as C_n.

2.2 Eigenvalues and eigenvectors of matrix ***A***

Consider a graph with weights assigned to its nodes and edges. The nodal weight vector is,

$$\mathbf{NW} = [nw_i];\ i = 1,2,\ldots,n, \tag{1}$$

and edge weight vector is defined as:

$$\mathbf{EW} = [ew_{ij}];\ (i,j) = 1,,\ldots,n, \tag{2}$$

The adjacency matrix $\mathbf{A} = [a_{ij}]_{n\times n}$ of a weighted graph G, containing n nodes, is defined as:

$$a_{ij} = \begin{cases} ew_{ij} & \text{if } n_i \text{ is adjacent to } n_j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

For a non-weighted graph ew_{ij} should be replaced by unity.

Consider the eigenproblem as

$$\mathbf{A}\phi_i = \mu_i\phi_i \tag{4}$$

where μ_i is the eigenvalue and ϕ_i is the corresponding eigenvector. Since **A** is a symmetric real matrix, all it's eigenvalues are real and can be expressed as

$$\mu_1 \le \mu_2 \le \ldots \le \mu_{n-1} \le \mu_n \tag{5}$$

The largest eigenvalue μ_n is the root of the characteristic equation of **A** with multiplicity 1. The corresponding eigenvector ϕ_n is the only eigenvector with positive entries. This vector has attractive properties employed in geography and structural mechanics.

Gould [12] appears to have introduced the first important application on using the properties of ϕ_n in calculating the accessibility indices of cities. The city with the highest accessibility corresponds to the largest entry of ϕ_n.

Grimes et al. [13] used the node with smallest accessibility as a pseudo-peripheral node corresponding to the node with least entry of ϕ_n. Kaveh [14] used the properties of ϕ_n for complete nodal ordering.

*2.3 Eigenvalues and eigenvectors of matrix **L***

The entries of the weighted Laplacian matrix **L** of a weighted graph is defined as:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{6}$$

The entries of **L** are as follows:

$$l_{ij} = \begin{cases} -ew_{ij} = -ew_{ji} & \text{if nodes } n_i \text{ and } n_j \text{ are adjacent} \\ ew_i = \sum_{j=1}^{D_i} ew_{ij} & \text{for } i = j \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

In this relation ew_{ij} is the weight of the edge e_{ij}, and D_i is the degree of the node n_i. For a non-weighted graph, the degree matrix $\mathbf{D} = [d_{ij}]_{n\times n}$ is a diagonal matrix of node degrees. Here, the *i*th diagonal entry d_{ii} is equal to the degree of the node i. Therefore, the entries of **L** are as:

$$l_{ij} = \begin{cases} -1 & \text{if node } n_i \text{ is adjacent to } n_j \\ \deg(n_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Consider the following eigenproblem:

$$\mathbf{L}\mathbf{v}_i = \lambda_i \mathbf{v}_i, \tag{9}$$

where λ_i is the eigenvalue and $\mathbf{v}_i$ is the corresponding eigenvector. As for **A,** all the eigenvalues of **L** are real. It can be shown that matrix **L** is a positive semi-definite matrix with

$$0 = \lambda_1 \le \lambda_2 \le \ldots \le \lambda_n \tag{10}$$

and

$$\mathbf{v}_1^t = \{1,\ 1,\ \dots\ ,1\}$$

The second eigenvalue λ_2 and the corresponding eigenvector $\mathbf{v}_2$ has attractive properties. Fiedler [6] has investigated various properties of λ_2. This eigenvalue is known as the *algebraic connectivity* of a graph, and the corresponding eigenvector $\mathbf{v}_2$ is known as the *Fiedlers vector*.

Mohar [7] has applied $(\lambda_2,\mathbf{v}_2)$ to different problems such as graph partitioning and ordering. Paulino et al. [15] used $\mathbf{v}_2$ for element ordering and nodal numbering.

Pothen et al. [16], Simon [17], Seale and Topping [18], and Kaveh and Davaran [19] and Kaveh and Rahimi Bondarabady [20-21] have used the properties of $\mathbf{v}_2$, for partitioning graphs. However, for calculating λ_2 when the entire model is considered, a fair amount of computational time and storage space is required. In this paper, for regular structural models, this goal is achieved by a far simple and more efficient analytical method.

3. GRAPH PRODUCTS

3.1 Cartesian Product of Two Graphs

Many structures have regular patterns and can be viewed as the Cartesian product of a number of simple graphs. These subgraphs, which are used in the formation of a model, are called the *generators* of that model.

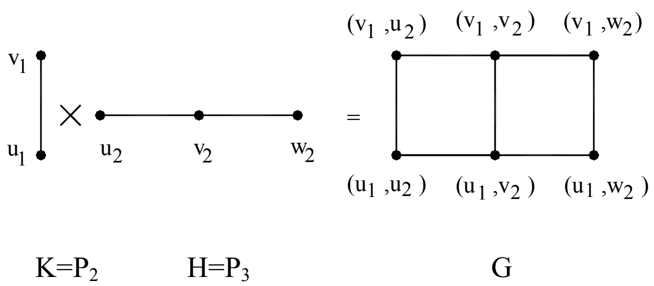
The simplest Boolean operation on a graph, is the Cartesian product K×H introduced by Sabidussi [22]. The Cartesian product is a Boolean operation G = K×H in which for any two nodes $u = (u_1,u_2)$ and $v = (v_1,v_2)$ in N(K)×N(H), the member uv is in E(G) whenever,

$$u_1 = v_1 \text{ and } u_2v_2 \in E(H), \tag{11a}$$

or

$$u_2 = v_2 \text{ and } u_1v_1 \in E(K). \tag{11b}$$

As an example, the Cartesian product of $K = P_2$ and $H = P_3$ is shown in Figure 1.



(a) (b)

Figure 1. The Cartesian product of two simple graphs.

Example: In this example, the Cartesian product $C_7 \times P_5$ of the path graph with 5 nodes denoted by P_5 and a cycle graph shown by C_7 is illustrated in Figure 2.

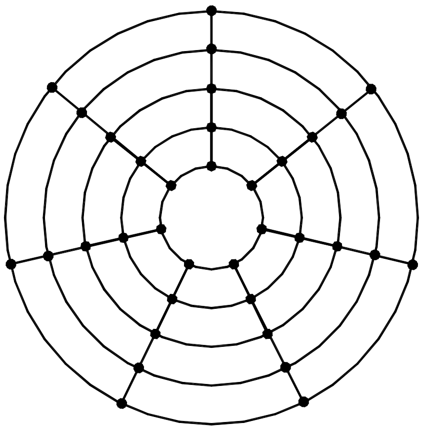


Figure 2. Representations of $C_7 \times P_5$

3.2 Strong Cartesian Product of Two Graphs

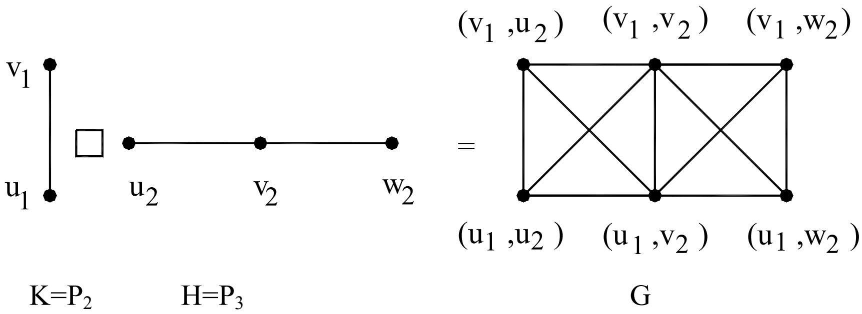
This is another Boolean operation, known as the *strong Cartesian product*. The strong Cartesian product is a Boolean operation $G = K \boxtimes H$ in which, for any two nodes $u=(u_1,u_2)$ and $v=(v_1,v_2)$ in $N(K) \times N(H)$, the member uv is in $E(G)$ if :

$$u_1 = u_2 \text{ and } v_1v_2 \in M(K) \text{ is in } E(H), \tag{12a}$$

$$v_1 = v_2 \text{ and } u_1u_2 \in M(K) \text{ is in } E(K), \tag{12b}$$

$$u_1u_2 \in E(K) \text{ and } v_1v_2 \in E(H) \tag{12c}$$

As an example, the strong Cartesian product of $K = P_2$ and $H = P_3$ is shown in Figure 3.



(a) Generators. (b) $G = K \boxtimes H$.

Figure 3. The strong Cartesian product of two simple graphs

Example: In this example, the strong Cartesian product $P_7 \boxtimes P_5$ of a path graph with 7 nodes, denoted by P_7 , and the path graph P_5, is illustrated in Figure 4.

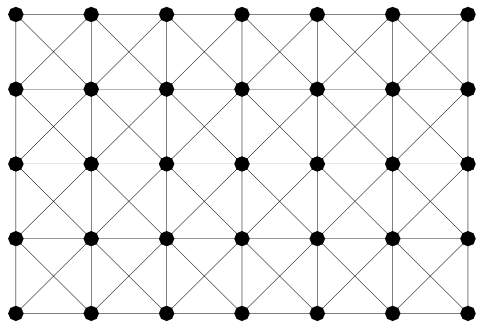


Figure 4. Strong product representation of $P_7 \boxtimes P_5$

3.3 Direct Product of Two Graphs

This is another Boolean operation known as the *direct product* introduced by Weichsel [23], who called it the *Kronecker Product*. The direct product is a Boolean operation G = K*H in which for any two nodes $u=(u_1,u_2)$ and $v=(v_1,v_2)$ in $N(K)\times N(H)$, the member uv is in E(G) if:

$$u_1v_1 \in M(K) \text{ and } u_2v_2 \in E(H). \tag{13}$$

As an example, the direct product of $K=P_2$ and $H=P_3$ is shown in Figure 5.

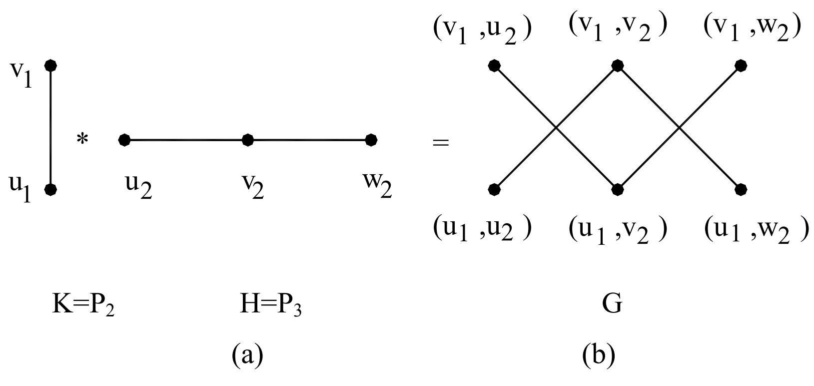


Figure 5. The direct product of two simple graphs

Example: The direct product P_7*P_5 of the path graph P_7 and path graph P_5 is illustrated in Figure 6.

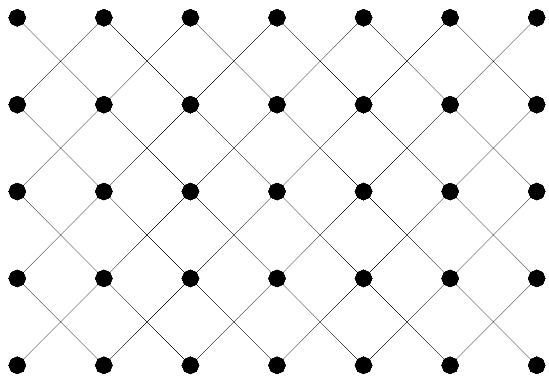


Figure 6. Direct product representation of $P_7 * P_5$

3.4 Kronecker Product

The *Kronecker product* of two matrices **A** and **B**, is the matrix we get by replacing the *ij*-th entry of **A** by a_{ij}**B**, for all i and j.

As an example,

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a & b & a & b \\ c & d & c & d \\ a & b & 0 & 0 \\ c & d & 0 & 0 \end{bmatrix} \tag{14}$$

where entry 1 in the first matrix has been replaced by a complete copy of the second matrix.

The Kronecker product has the property that if **B**, **C**, **D**, and **E** are four matrices, such that **BD** and **CE** exists, then:

$$(\mathbf{B}\otimes\mathbf{C})(\mathbf{D}\otimes\mathbf{E}) = \mathbf{BD}\otimes\mathbf{CE}. \tag{15}$$

Thus, if u and v are vectors of the correct dimensions, then:

$$(\mathbf{B}\otimes\mathbf{C})(\mathbf{u}\otimes\mathbf{v}) = \mathbf{Bu}\otimes\mathbf{Cv}. \tag{16}$$

If u and v are eigenvectors of B and C, with eigenvalues λ and μ, respectively, then,

$$\mathbf{Bu}\otimes\mathbf{Cv} = \lambda\mu\mathbf{u}\otimes\mathbf{v}, \tag{17}$$

Whence **u**⊗**v** is an eigenvector of **B**⊗**C** with eigenvalue $\lambda\mu$.

4. A UNIFIED APPROACH FOR EIGENVALUES OF GRAPH PRODUCTS

Consider a block tri-diagonal matrix in the following form:

$$\mathbf{M}_{mn} = \begin{bmatrix} \mathbf{A} & \mathbf{B} & & & & & & \\ \mathbf{B} & \mathbf{C} & \mathbf{B} & & & & & \\ & \mathbf{B} & \mathbf{C} & \mathbf{B} & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & \cdot & \cdot & \cdot & \\ & & & & \mathbf{B} & \mathbf{C} & \mathbf{B} & \\ & & & & & \mathbf{B} & \mathbf{C} & \mathbf{B} \\ & & & & & & \mathbf{B} & \mathbf{A} \end{bmatrix}, \tag{18}$$

where **A**, **B** and **C** are m×m matrix blocks. The matrix $\mathbf{M}_{mn}$ contains n blocks in each row and n blocks in each column. A matrix $\mathbf{M}_{mn}$ in the form of Eq. (18) will be denoted by $\mathbf{M}_{mn} = \mathbf{F}(\mathbf{A}_m,\mathbf{B}_m,\mathbf{C}_m)_{mn}$. Now we study various forms of $\mathbf{M}_{mn}$.

4.1 FORM 1 (for adjacency matrices)

This form corresponds to the adjacency matrices of three groups of graph products, namely Cartesian, Strong Cartesian and Direct products. Here, it is assumed that weights are associated with the nodes of the graph.

In this form,

$$\mathbf{M}_{mn} = \mathbf{F}(\mathbf{A}_m,\mathbf{B}_m,\mathbf{A}_m)_{mn}, \tag{19}$$

where

$$\mathbf{A}_m = \mathbf{F}(a,b,a)_m \text{ and } \mathbf{B}_m = \mathbf{F}(c,d,c)_m\,. \tag{20}$$

The small characters are numbers, and the capital characters are matrices.

Consider $\mathbf{T}_k = \mathbf{F}(0,1,0)_k$ with eigenvalues λ_k, and denote the unit matrix by $\mathbf{I}_k$, where k is the dimension of the square matrices $\mathbf{T}_k$ and $\mathbf{I}_k$. Using the properties of the Kroneker products from linear algebra, the matrix $\mathbf{M}_{mn}$ can be decomposed as:

$$\mathbf{M}_{mn} = \mathbf{I}_n \otimes \mathbf{A}_m + \mathbf{T}_n \otimes \mathbf{B}_m. \tag{21}$$

Substituting

$$\mathbf{A}_m = (a\mathbf{I}_m + b\mathbf{T}_m) \text{ and } \mathbf{B}_m = (c\mathbf{I}_m + d\mathbf{T}_m), \tag{22}$$

leads to:

$$\mathbf{M}_{mn} = a\mathbf{I}_n \otimes \mathbf{I}_m + b\mathbf{I}_n \otimes \mathbf{T}_m + c\mathbf{T}_n \otimes \mathbf{I}_m + d\mathbf{T}_n \otimes \mathbf{T}_m. \tag{23}$$

Therefore:

$$\lambda = a + b\lambda_m + c\lambda_n + d\lambda_m\lambda_n. \tag{24}$$

For graph with no weights, the Cartesian product, strong Cartesian product and direct product have coefficients a, b, c and λ as provided in Table 1.

Table 1. The coefficients of λ.

Type of Product	**a**	**b**	**c**	**d**	**λ**
Cartesian	0	1	1	0	$\lambda = \lambda_m + \lambda_n$
Strong Cartesian	0	0	0	1	$\lambda = \lambda_m\lambda_n$.
Direct	0	1	1	1	$\lambda = \lambda_m + \lambda_n - \lambda_m\lambda_n$.

Path Graphs: For a path graph P_m with m nodes, we have the following special case,

$$\mathbf{M}_{mn} = \mathbf{F}(\mathbf{A}_m,\mathbf{B}_m,\mathbf{A}_m)_{mn}, \tag{25}$$

where $$\mathbf{A}_m = \mathbf{F}(0,b,0)_m \text{ and } \mathbf{B}_m = \mathbf{F}(c,d,c)_m. \tag{26}$$

The matrix M_{mn} can be decomposed as:

$$\begin{aligned}\mathbf{M}_{mn} &= \mathbf{I}_n \otimes \mathbf{A}_m + \mathbf{T}_n \otimes \mathbf{B}_m \\ &= b\mathbf{I}_n \otimes \mathbf{T}_m + \mathbf{T}_n \otimes (c\mathbf{I}_m + d\mathbf{T}_m) \\ &= b\mathbf{I}_n \otimes \mathbf{T}_m + c\mathbf{T}_n \otimes \mathbf{I}_m + d\mathbf{T}_n \otimes \mathbf{T}_m.\end{aligned} \tag{27}$$

Therefore:

$$\lambda = b\lambda_m + c\lambda_n + d\lambda_m\lambda_n. \tag{28}$$

Cycle Graphs: For a cycle graph C_m, the matrix $\mathbf{M}_{mn}$ is a tri-diagonal matrix similar to that of the path graph, with the difference of A_m and B_m having an entry p in the two corners as:

$$\mathbf{F}=\begin{bmatrix} * & * & & & & & & & p \\ * & * & * & & & & & & \\ & * & * & * & & & & & \\ & & \cdot & \cdot & \cdot & & & & \\ & & & \cdot & \cdot & \cdot & & & \\ & & & & \cdot & \cdot & \cdot & & \\ & & & & & * & * & * & \\ & & & & & & * & * & * \\ p & & & & & & & * & * \end{bmatrix} \tag{29}$$

where p is a number.

4.2 FORM 2 (for Laplacian matrices)

This form appears in Laplacian matrices of weighted graphs for Cartesian products, and strong Cartesian and direct products after addition of boundary edges, Ref. [11].

Path Graphs: For a path graph we have:

$$\mathbf{M}_{mn}=\mathbf{F}(\mathbf{A}_m,\mathbf{B}_m,\mathbf{C}_m)_{mn}, \tag{30}$$

where

$$\mathbf{A}_m=\mathbf{F}(a_1,b_1,c_1),\ \mathbf{B}_m=\mathbf{F}(a_2,b_2,c_2),\ \mathbf{C}_m=\mathbf{F}(a_3,b_3,c_3) \tag{31}$$

For this form we have $\mathbf{A}=\mathbf{B}+\mathbf{C}$ and $a_i = b_i + c_i$ (i=1,2,3). Assuming $\mathbf{T}_k = \mathbf{F}(1,-1,2)_k$ with λ_k being the eigenvalues of $\mathbf{T}_k$, we have:

$$\mathbf{M}_{mn}=\mathbf{I}_n\otimes(\mathbf{A}+\mathbf{B})_m+\mathbf{T}_n\otimes(-\mathbf{B})_m \tag{32}$$

But $(\mathbf{A}+\mathbf{B})_m = (a_1+a_2+b_1+b_2)\mathbf{I}_m - (b_1+b_2)\mathbf{T}_m$, and $\mathbf{B}_m = (a_2+b_2)\mathbf{I}_m - b_2\mathbf{T}_m$. (33)

Therefore,

$$\mathbf{M}_{mn}=(a_1+a_2+b_1+b_2)\mathbf{I}_{mn}-(b_1+b_2)\mathbf{I}_n\otimes\mathbf{T}_m-(a_2+b_2)\mathbf{T}_n\otimes\mathbf{I}_m+b_2\mathbf{T}_n\otimes\mathbf{T}_m \tag{34}$$

and

$$\lambda=(a_1+a_2+b_1+b_2)-(b_1+b_2)\lambda_m-(a_2+b_2)\lambda_n+b_2\lambda_n\lambda_m \tag{35}$$

For graph without weights, the Cartesian product, strong Cartesian product and direct product the coefficients $\mathbf{A}_m$, $\mathbf{B}_m$, $\mathbf{C}_m$ and λ are provided in Table 2.

Table 2. The coefficients of λ.

Product	$\mathbf{A_m}$	$\mathbf{B_m}$	$\mathbf{C_m}$	λ
Cartesian	$\mathbf{F}(2,-1,3)$	$\mathbf{F}(-1,0,-1)$	$\mathbf{F}(3,-1,4)$	$\lambda = \lambda_m + \lambda_n$
Strong Cartesian	$\mathbf{F}(3,-1,4)$	$\mathbf{F}(-1,-1,0)$	$\mathbf{F}(4,0,4)$	$\lambda = 2\lambda_m + 2\lambda_n - \lambda_m\lambda_n$.
Direct	$\mathbf{F}(5,-2,2)$	$\mathbf{F}(-2,-1,-1)$	$\mathbf{F}(7,-1,8)$	$\lambda = 3\lambda_m + 3\lambda_n - \lambda_m\lambda_n$.

Note: For weighted graphs, the weight of the added boundary edge should be considered as the weight of the diagonal (bracing) edges.

Cycle Graphs: For cycle graphs, λ_m corresponding to a $\mathbf{T}_m$ contains additional entries -1 in the two far corners.

Once the eigenvalues are found, the corresponding eigenvectors can be calculated. However, this can be done much simpler considering that the eigenvectors of G are the Kronecker product of the eigenvectors of K and H, i.e. $\mathbf{w}_k = \mathbf{u}_i \otimes \mathbf{v}_j$, where $\mathbf{w}_k$, $\mathbf{u}_i$ and $\mathbf{v}_j$ are the eigenvectors of G, K and H, respectively.

Example: Consider the Cartesian product of P_4 and P_5. Let $G = P_4 \times P_5$ be a weighted graph with horizontal edges having weight 2 and the vertical ones with weight 4. In this case, the adjacency matrix **A** will have the following form:

$$\mathbf{M}_{mn} = \mathbf{F}(\mathbf{A_5},\mathbf{B_5},\mathbf{A_5})_{\mathbf{54}},\ \mathbf{A_5} = \mathbf{F}(0,2,0),\ \mathbf{B_5} = \mathbf{F}(3,4,3)_{\mathbf{5}}$$

which is the same as Form I, and with λ_4 and λ_5 being taken as the eigenvalues of P_4 and P_5, respectively, we have

$$\lambda = 2\lambda_5+3\lambda_4+4\lambda_5\lambda_4;\ \ \lambda_n = 2\cos\frac{k\pi}{n+1}\ \ (k=1,\ldots,n),$$

and

$$\lambda_{min} = -8.3182 \text{ and } \lambda_{max} = 8.3182.$$

For the Laplacian matrix **L**, λ_4 and λ_5 correspond to the Laplacian of P_4 and P_5, and

$$\lambda_n = 2-2\cos\frac{k\pi}{n}\ \ (k=0,\ldots,n-1)$$

with

$$\mathbf{M}_{54} = \mathbf{F}(\mathbf{A}_5,\mathbf{B}_5,\mathbf{C}_5)_{54},\ \mathbf{A}_5 = \mathbf{F}(5,-2,7)_5,\ \mathbf{B}_5 = \mathbf{F}(-3,0,-3)_5,\ \mathbf{C}_5 = \mathbf{F}(8,-2,10)_5$$

It can be observed that $\mathbf{A}_5 = \mathbf{B}_5 + \mathbf{C}_5$ and $a_i = b_i + c_i$ (for i=1,2,3), which are the properties corresponding to Form II. Therefore:

$$\lambda=[(5)+(-3)+(-2)+(0)]-[(-2)+(0)]\lambda_5-[(-3)+(0)]\lambda_4+(0)\lambda_5\lambda_4=2\lambda_5+3\lambda_4$$

leading to

$$\lambda_2 = 2(0.3820) + 3(0) = 0.7639$$

If one considers $C_5 \times P_4$, then λ_5 corresponding to C_5 should be employed. Then we will have numbers in the corner entries of the matrix and a similar method can be used.

For strong Cartesian product of P_4 and P_5, the adjacency matrix has Form I, however, for Laplacian matrix none of the forms discussed will be observed. For this case, edges are added to the boundary nodes (or the weights of the boundary edges are doubled) in order to construct regular graphs. After this operation, Form II is produced and the calculations are performed as before. For direct product, a similar operation can be employed for the Laplacian matrix **L**.

In this paper, no comparison is made, since the present methods are applicable only to regular models. Due to the analytical nature of these approaches, the computational time is far less then the standard methods for calculating eigenvalues. For applications of the present methods, the reader may refer to Refences [11,24].

5. CONCLUDING REMARKS

The unified method presented for calculating the eigenvalues of the adjacency and Laplacian matrices of three different graph products, provides an efficient approach for calculating the eigenvalues of adjacency and Laplacian matrices of weighted and non-weighted graphs. The eigensolution of graphs has many applications in computational mechanics. Examples of such applications are nodal and element ordering for bandwidth, profile and frontwidth optimization, graph partitioning, and subdomaining of finite element models [13-21]. The present forms are also effective tools for calculating eigenvalues and eigenvectors of matrices arising from numerical methods for differential equations applied to structural mechanics problems. Such applications are presented in Refs. [11,24].

REFERENCES

1. Kaveh, A. *Structural Mechanics: Graph and Matrix Methods*, 3rd edition, Research Studies Press, Somerset, UK, 2004.
2. Kaveh, A. *Optimal Structural Analysis*, 2nd edition, John Wiley, UK, 2006.
3. Biggs, N.L. *Algebraic Graph Theory*, Cambridge University Press, 2nd edition, Cambridge, 1993.
4. Cvetković, D.M., Doob, M. and H. Sachs, H. *Spectra of Graphs, Theory and Applications,* Academic Press, 1980.

5. Godsil C. and Royle, G. *Algebraic Graph Theory*, Springer Verlag, New York, 2001.
6. Fiedler, M. Algebraic connectivity of graphs, *Czech. Math. J.*, 1973; 23:298-305.
7. Mohar, B. The Laplacian spectrum of graphs, entitled Graph Theory, Combinatorics and Applications, edit. Y. Alavi et al., Vol. 2, John Wiley, NY, 1991, pp 871-898.
8. Pothen, A. Simon, H. and Liou, K.P. Partitioning sparse matrices with eigenvectors of graphs, *SIAM J. Matrix Anal. Appl*., 1990; 11:430-452.
9. Topping, BHV and Sziveri, J. Parallel subdomain generation method, Proc. Civil-Comp, Edinburgh, UK, 1995.
10. Kaveh, A. and Rahami, H. A new spectral method for nodal ordering of regular space structures, *Finite Elements in Analysis and Design*, Nos 13/14,2004; 40:1931-1945.
11. Kaveh, A. and Rahami, H. An efficient method for decomposition of regular structures using graph products, 2004; 61:1797-1808.
12. Gould, P. The geographical interpretation of eigenvalues, *Transactions of the Institute of British Geographer*, 1967; 42:53-58.
13. Grimes, R.G., Pierce, D.J. and Simon, H.D. A new algorithm for finding a pseudo-peripheral node in a graph, *SIAM Journal of Analysis and Applications*, **11**(1990) 323-334.
14. Kaveh, A. Algebraic and topological graph theory for ordering, *ZAMM*, **71**(1991) T739-T742.
15. Paulino, G.H., Menezes, IFM. Gattass, M. and Mukherjee, S. Node and element resequencing using Laplacian of a finite element graph, Part-I-general concepts and algorithms, *International Journal of Numerical Methods in Engineering*, **37**(1994) 1511-1530.
16. Pothen, A., Simon H. and Liou K. P., Partitioning sparse matrices with eigenvectors of graphs, *SIAM J. Matrix Anal. Appl*., **11**(1990) 430-452.
17. Simon, H.D. Partitioning of unstructured problems for parallel processing, *Computing in System Engineering*, **2**(1991**)** 135-148.
18. Seale C. and Topping, B.H.V. Parallel implementation of recursive spectral bisection. Proceedings of Civil Comp 95, Edinburgh, UK, 1995, pp. 459-473.
19. Kaveh, A. and Davaran, A. Spectral bisection of adaptive finite element meshes for parallel processing, *Computers and Structures*, **70**(1999) 315-327.
20. Kaveh, A. and Rahimi Bondarabady, H.A. Spectral trisection of finite element models, Computers and Structures, *International Journal of Numerical Methods for Heat and Fluid Flow*, **11**(2001) 358-370.
21. Kaveh, A. and Rahimi Bondarabady, H.A. A multi-level finite element nodal ordering using algebraic graph theory, *Finite Elements in Analysis and Design*, **38**(2002) 245-261.
22. Sabidussi, G., Graph Multiplication, *Math. Z*., **72**(1960) 446-457.
23. Weichsel, P.M., The Kronecker product of graphs, *Proc. Amer. Math. Soc*., **13**(1962) 47-52.
24. Kaveh, A. and Rahami, H., New canonical forms for analytical solution of problems in structural mechanics, *Int. J. Numer. Meths Engng*, 2006.