RESEARCH REVIEW JOURNALS
Volume-04
Issue-03
March -2019

ISSN: 2455-3085 (Online)
RESEARCH REVIEW International Journal of Multidisciplinary
www.rrjournals.com [UGC Listed Journal]

A Study of Mathematical Formulation on Computational Graph Theory

[1]Anant Phirke & [2]Dr. Ashwini Kumar Nagpal

[1]*Research Scholar OPJS University Churu Rajasthan*
[2]*Professor OPJS University Churu Rajasthan*

ARTICLE DETAILS

Article History
Published Online: 13 March 2019

Keywords
Mathematical formulation, computational, graph theory, graph products, structural models.

ABSTRACT

The first step in the analysis of a structure is to generate its configuration. Different means are available for this purpose. The use of graph products is an example of such tools. In this paper, the use of product graphs is extended for the formation of different types of structural models. Here weighted graphs are used as the generators and the connectivity properties of different models are expressed in terms of the properties of their generators through simple algebraic relationships. In this paper by using graph product concepts and spatial structured matrices, a new algebraic closed form is proposed for mathematical formulation and presentation of structures. For clarification some examples are included.

1. Introduction

Data generation is the first step in the analysis of every structure. Configuration processing of large scale problems without automatic approaches can be erroneous and occasionally impossible. Formex configuration processing is one such a means introduced by Nooshin [1] and further developed by Nooshin et al. [2] and Nooshin and Disney [3]. Similar methods are developed based on set theory by Behravesh et al. [4]. Kaveh applied graph theory for this formation [5] (see also Kaveh et al. [6]). The use of product graphs in structural mechanics is suggested in [7, 8] and application of the corresponding concepts utilizing the directed and looped generators is due to Kaveh and Koohestani [9], weighted graph products by Kaveh and Nouri [10] and weighted triangular and circular graph products employed by Kaveh and Beheshti [11].

There are many other references in the field of data generation; however, most of them are prepared for specific classes of a problem. For example, many algorithms have been developed and successfully implemented on mesh or grid generation; a complete review of which may be found in a paper by Thacker [12] and in the books by Thompson et al. [13], Liseikin [14], and Topping et al. [15].

In this paper the configuration processing of regular structures is considered. A structure is called regular if it can be considered as the product of two or three subgraphs (generators). The weighted graph products developed in [10] and their application are extended. Weighted paths and cycles are considered as the generators, and it is shown that many such product graphs can algebraically be expressed by simple relationships and a new algebraic closed form proposed for mathematical formulation and presentation of structures. Once this is done, then the existing methods can be applied to eigensolution and analysis of such structures. However, this paper is limited to the generalization of product graphs for configuration processing of space structures. The methods of this paper can easily be adopted in the mesh generation of the finite element models.

2. Graph Theory

A graph consists of a set of elements, $N(S)$, called nodes and a set of elements, $M(S)$, called members, together with a relation of incidence which associates two distinct nodes with each member, known as its ends. If weights are assigned to the members and nodes of a graph, then it becomes a weighted graph, (Figure 1). Two nodes of a graph are called adjacent if these nodes are the end nodes of a member. A member is called incident with a node if that node is an end node of the member. The degree of a node is the number of members incident with that node. A subgraph S_i of a graph is a graph for which $N(S_i) \subseteq N(S)$ and $M(S_i) \subseteq M(S)$, and , and each member of has the same ends as in . A path graph is a simple connected graph with $N(P) = M(P) + 1$ that can be drawn in a way that all of its nodes and members lie on a single straight line. A path graph with nodes is denoted by , and a weighted path is shown by . A cycle graph . P_n, is a simple connected graph with identical number of nodes and members that can be drawn so that all of its nodes and members lie on a circle. A cycle graph with nodes is shown by , and a weighted cycle is denoted by . Examples of these graphs are shown in Figure 1. For further definitions the reader may refer to Kaveh.

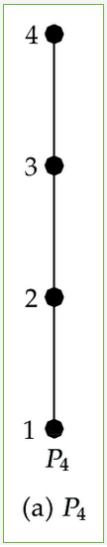

Figure 1: Examples of simple and weighted graphs

3. Algebraic representation of path and cycles

Most of the space structures can be viewed as the product of some weighted paths and cycles. Therefore in this section some simple mathematical relationships are presented for defining such generators.

1. Weighted Path

The adjacency matrix of a path in general can be expressed as

$$\operatorname{adj}(P_n w) = \begin{bmatrix} W_1 & W_{1,2} & & & \\ W_{2,1} & W_2 & W_{2,3} & & \\ & \ddots & \ddots & \ddots & \\ & & W_{n-1,n-2} & W_{n-1} & W_{n-1,n} \\ & & & W_{n,n-1} & W_n \end{bmatrix}_{n\times n} = \begin{bmatrix} \ddots & \ddots & \ddots & \\ & L & D & U & \\ & & \ddots & \ddots & \ddots \end{bmatrix},$$

$$L = \begin{bmatrix} W_{2,1} \\ W_{3,2} \\ \vdots \\ W_{n-1,n-2} \\ W_{n,n-1} \end{bmatrix}_{(n-1)\times 1}, \quad D = \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_{n-1} \\ W_n \end{bmatrix}_{n\times 1}, \quad U = \begin{bmatrix} W_{1,2} \\ W_{2,3} \\ \vdots \\ W_{n-2,n-1} \\ W_{n-1,n} \end{bmatrix}_{(n-1)\times 1}, \tag{1}$$

where the weights are divided into 3 groups, ,*L* , *D*and *U*.

Using this definition a weighted path, in general, can be expressed as

$$P_n w \begin{bmatrix} L^T \\ D^T \\ U^T \end{bmatrix}. \tag{2}$$

2. Weighted Cycle

The adjacency matrix of a weighted cycle can similarly be expressed as

$$\operatorname{adj}(C_n w) = \begin{bmatrix} W_1 & W_{1,2} & & & W_{1,n} \\ W_{2,1} & W_2 & W_{2,3} & & \\ & \ddots & \ddots & \ddots & \\ & & W_{n-1,n-2} & W_{n-1} & W_{n-1,n} \\ W_{n,1} & & & W_{n,n-1} & W_n \end{bmatrix}_{n\times n} = \begin{bmatrix} \ddots & \ddots & \ddots & \\ & L & D & U & \\ & & \ddots & \ddots & \ddots \end{bmatrix} \quad (3)$$

$$L = \begin{bmatrix} W_{2,1} \\ W_{3,2} \\ \vdots \\ W_{n,n-1} \\ W_{1,n} \end{bmatrix}_{n\times 1}, \quad D = \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_{n-1} \\ W_n \end{bmatrix}_{n\times 1}, \quad U = \begin{bmatrix} W_{1,2} \\ W_{2,3} \\ \vdots \\ W_{n-1,n} \\ W_{n,1} \end{bmatrix}_{n\times 1},$$

where the weights are also divided into 3 groups ,*L* ,*D* and *U*.

Considering these, a weighted cycle, in general, can be shown as

$$C_n w \begin{bmatrix} L^T \\ D^T \\ U^T \end{bmatrix}. \quad (4)$$

3. Unit and Zero Vectors

The unit vector is defined as the following.

$E_n$is an *n* by 1 vector with all entries being 1. In addition $E_n(i)$ is a vector of the same dimension with all entries as 1 except the entry at *i*th row which is zero:

$$E_n = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{n\times 1}, \quad E_n(i) = \begin{bmatrix} 1 \\ \vdots \\ 0 \\ \vdots \\ 1 \end{bmatrix}_{n\times 1} \longrightarrow i^{'}\text{th row.} \quad (5)$$

The zero vector is defined as the following.

$O_n$is an *n* by 1 vector with all entries being 0. In addition $O_n(i)$ is a vector of the same dimension with all entries as 0 except the entry at the *i*th row which is 1:

$$O_n = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{n\times 1}, \quad O_n(i) = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}_{n\times 1} \longrightarrow i^{'}\text{th row.} \quad (6)$$

4. Extension of the Zero and Unit Vectors

In this section the zero and unit vectors are extended to represent [*L*,*D*,*U*] in an efficient manner.

If we want to create a vector with some entries as 1 and the remaining also as 0, we use the following expression:

$$O_n(1,2,i,k) = O_n(1) + O_n(2) + O_n(i) + O_n(k), \quad i,k \le n. \tag{7}$$

If we want to create a vector with the $I, i+k, i+2k, \cdots$ as 1 and the remaining entries as 0, we use the following expression:

$$O_n(i:k) = O_n(i) + O_n(i+k) + O_n(i+2k) + O_n(i+3k) + \cdots + O_n(m), \quad n-k \le m \le n. \tag{8}$$

For creating a vector with the $I,\ i+k,\ i+2k, \ldots, i+m*k$ as 1 and the remaining entries as 0, we use the following expression:

$$O_n(i:k:m) = O_n(i) + O_n(i+k) + O_n(i+2k) + O_n(i+3k) + \cdots + O_n(i+mk), \quad i+mk \le n \tag{9}$$

For creating a vector with the $(I, j, \ldots), (i+k, j+k, \ldots), (i+2k, j+3k, \ldots), \ldots,$ and $(i+mk, j+mk, \ldots)$ as 1 and the remaining entries as 0, we use the following expression:

$$\begin{aligned} &O_n(i, j, \ldots : k : m) \\ &\quad = [O_n(i) + O_n(j) + \cdots] + [O_n(i+k) + O_n(j+k) + \cdots] \\ &\qquad + [O_n(i+2k) + O_n(j+2k) + \cdots] + \cdots + [O_n(i+mk) + O_n(j+mk) + \cdots]. \end{aligned} \tag{10}$$

In general case the following relation exists between the zero and unit vectors:

$$E_n(i) = E_n - O_n(i). \tag{11}$$

As an example the weighted graphs shown in Figure 2 are expressed in algebraic form.

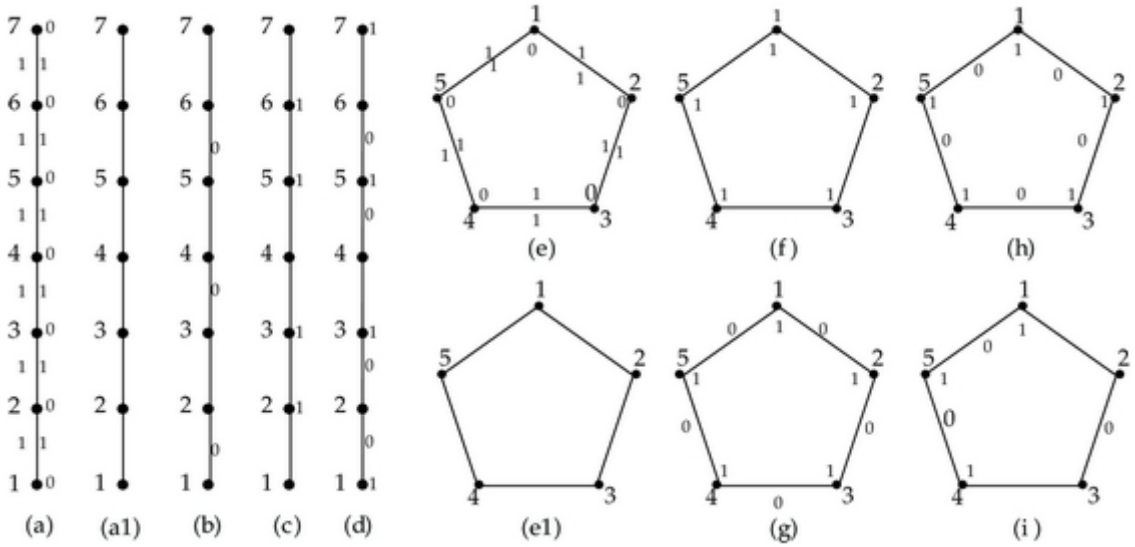

Figure 2: Some weighted graphs

In compact algebraic representation the difference between a simple and a weighted graph is illustrated. As an example, for Figures 2(a) and 2(a1), which are both simple paths, a is weighted and a1 is simple, the algebraic representations are as follows:

$$(\mathrm{a}) \Rightarrow P_7 w \begin{bmatrix} L : E_6^T \\ D : O_7^T \\ U : E_6^T \end{bmatrix}, \qquad (\mathrm{a}_1) \Rightarrow P_7. \tag{12}$$

For the weighted case $L, D,$ and U are E_6^I, O_7^I, E_6^I, respectively.

In Figure 2(i) a weighted cycle is shown, where L, D, and U are $E_5^T(4,5), O_5^T(1,4,5), E_5^T(2)$, respectively. In the following weighted paths and cycles of Figure 2 are represented algebraically:

$$(a, a_1) \Rightarrow P_7 = P_7 w \begin{bmatrix} E_6^T \\ O_7^T \\ E_6^T \end{bmatrix}, \quad (b) \Rightarrow P_7 w \left[L : E_6^T (1:2) \right], \quad (c) \Rightarrow P_7 w \left[D : O_7^T (2,3:3) \right],$$

$$(d) \Rightarrow P_7 w \begin{bmatrix} L : E_6^T (1,2:3) \\ D : O_7^T (1:2) \end{bmatrix}, \quad (e, e_1) \Rightarrow C_5 = C_5 w \begin{bmatrix} E_5^T \\ O_5^T \\ E_5^T \end{bmatrix}, \quad (f) \Rightarrow C_5 w \left[D : E_5^T \right],$$

$$(g) \Rightarrow C_5 w \begin{bmatrix} L : O_5^T \\ D : E_5^T \end{bmatrix}, \quad (h) \Rightarrow C_5 w \begin{bmatrix} D : E_5^T \\ U : O_5^T \end{bmatrix}, \quad (i) \Rightarrow C_5 w \begin{bmatrix} E_5^T (4,5) \\ O_5^T (1,4,5) \\ E_5^T (2) \end{bmatrix}. \tag{13}$$

4. Conclusions

In this paper the graph products and their applications in configuration processing are extended. Topology of a structure is viewed as the product of two weighted sub graphs like paths and/or cycles as its generators. The paths and cycles are formulated in a mathematical form, and the configuration of a space structure is expressed as different products of these weighted sub graphs as one expression. In the presented method the topological information of space structures can be stored as simple algebraic relationships. More complex configurations can be formulated using different graph theory operators and new conditions can be added to the domains of the products. The application of the introduced products of weighted graphs can also be extended to the mesh generation of finite element models.

References

1. H. Nooshin, "Algebraic representation and processing of structural configurations," Computers and Structures, vol. 5, no. 2-3, pp. 119–130, 1975. View at Publisher · View at Google Scholar · View at Zentralblatt MATH · View at Scopus
2. H. Nooshin, P. Disney, and C. Yamamoto, Formian, Multi-Science Publishers, Essex, UK, 1993.
3. H. Nooshin and P. Disney, "Formex configuration processing III," International Journal of Space Structures, vol. 17, no. 1, pp. 1–50, 2002. View at Publisher · View at Google Scholar · View at Scopus
4. A. Behravesh, A. Kaveh, M. Nani, and S. Sabet, "A set theoretical approach for configuration processing," Computers & Structures, vol. 30, no. 6, pp. 1293–1302, 1988. View at Publisher · View at Google Scholar · View at Zentralblatt MATH · View at MathSciNet
5. A. Kaveh, "A graph-theoretical approach to configuration processing," Computers and Structures, vol. 48, no. 2, pp. 357–363, 1993. View at Publisher · View at Google Scholar · View at Zentralblatt MATH · View at Scopus
6. A. Kaveh, X. Jia, and Q. Weng, "Rotation as a general operation for configuration processing," Scientia Iranica, vol. 17, no. 2, pp. 131–140, 2010. View at Google Scholar · View at Zentralblatt MATH · View at Scopus
7. A. Kaveh, Optimal Structural Analysis, John Wiley & Sons, Somerset, UK, 2nd edition, 2006. View at MathSciNet
8. A. Kaveh and H. Rahami, "An efficient method for decomposition of regular structures using graph products," International Journal for Numerical Methods in Engineering, vol. 61, no. 11, pp. 1797–1808, 2004. View at Publisher · View at Google Scholar · View at Zentralblatt MATH · View at MathSciNet
9. A. Kaveh and K. Koohestani, "Graph products for configuration processing of space structures," Computers and Structures, vol. 86, no. 11-12, pp. 1219–1231, 2008. View at Publisher · View at Google Scholar · View at Scopus
10. A. Kaveh and M. Nouri, "Weighted graph products for configuration processing of planar and space structures," International Journal of Space Structures, vol. 24, no. 1, pp. 13–26, 2009. View at Publisher · View at Google Scholar · View at Scopus
11. A. Kaveh and S. Beheshti, "Weighted triangular and circular graph products for configuration processing," PeriodicaPolytechnica, vol. 56, pp. 1–9, 2012. View at Google Scholar
12. W. C. Thacker, "A brief review of techniques for generating irregular computational grids," International Journal for Numerical Methods in Engineering, vol. 15, no. 9, pp. 1335–1341, 1980. View at Publisher · View at Google Scholar · View at Zentralblatt MATH · View at Scopus
13. J. F. Thompson, Z. U. A. Warsi, and C. W. Mastin, Numerical Grid Generation, Foundations and Applications, Elsevier, Amsterdam, The Netherlands, 1985. View at MathSciNet
14. V. D. Liseikin, Grid Gneration Methods, Scientific Computation, Springer, Berlin, Germany, 1999. View at MathSciNet
15. B. H. V. Topping, J. Muylle, and P. Ivanyi, Finite Element Mesh Generation, Saxe Coburg, Pubn, UK, 2004.
16. W. Imrich and S. Klavžar, Product Graphs, Structure and Recognition, Wiley-Interscience, New York, NY, USA, 2000. View at MathSciNet

