Journal of Mechanisms, Transmissions, and Automation in Design

Discussion

An Application of the Linkage Characteristic Polynomial to the Topological Synthesis of Epicyclic Gear Trains (86-DET-66)

Jerry T. Pugh[1]. The author enumerates the permissible graphs of one-degree-of-freedom, epicyclic gear trains with up to six links. He shows that there are 80 nonisomorphic graphs from which all six-link gear trains can be constructed. He builds more complex graphs by adding to graphs previously enumerated.

In order to enumerate a graph with $n+1$ vertices, he adds a vertex, a turning-pair edge, and a gear edge to a graph with n vertices. The new turning-pair edge is connected between the added vertex and any one of the existing vertices; the geared edge is connected between the added vertex and any one of the remaining vertices. Some of the generated graphs are rejected due to violation of the fundamental rules associated with epicyclic gear trains [7, 11]. Isomorphic graphs are identified using a random number technique applied to the characteristic polynomial. These isomorphic graphs are eliminated in order to avoid duplication.

I suggest that by noting the symmetry of a graph, one can reduce the number of isomorphic graphs enumerated with the author's procedure.

First, it is convenient to represent the relations of a graph with various lists. For example, the graph of Fig. 1(*b*) can be represented with a list of edges (adjacent vertices) and a list of "geared" edges,

EDGES ((1 2) (2 3) (1 3)) GEARED-EDGES ((2 3))

Also, the symmetry of this graph can be represented by a list made up of disjoint sublists of symmetric vertices.

SYMMETRY ((1) (2 3))

Two vertices can be checked for symmetry by checking to see if they are members of the same sublist.

Second, the author's procedure is followed to enumerate a graph with $n+1$ vertices except that the symmetry of the n vertex graph is taken into consideration in order to avoid generating isomorphic graphs.

For example, consider the following steps to enumerate graphs with four vertices.

(*a*) Pick a starting vertex from the three vertex graph.
(*b*) If no vertices are left then quit.
(*c*) If the choosen vertex is symmetric with a vertex already picked as a starting vertex then go to (*a*).
(*d*) Form a gear pair with the starting vertex and vertex 4.
(*e*) Pick an ending vortex.
(*f*) If no vertices are left then go to (*a*)
(*g*) If the choosen ending vertex is the same as the starting vertex then go to (*d*).
(*h*) If the choosen vertex is symmetric with a vertex already picked then go to (*d*).
(*i*) Form a revolute pair betwen ending vertex and vertex 4.
(*j*) Go to (*a*).

Using the above procedure, the following three graphs are generated.

(1) EDGES ((1 2) (2 3) (1 3) (1 4) (4 2)) GEARED-EDGES ((2 3) (1 4))
(2) EDGES ((1 2) (2 3) (1 3) (2 4) (4 1)) GEARED-EDGES ((2 3) (2 4))
(3) EDGES ((1 2) (2 3) (1 3) (2 4) (4 3)) GEARED-EDGES ((2 3) (2 4))

Note that these graphs correspond to the three nonisomorphic graphs given by the author in Fig. 2(*a*), (*b*), and (*f*). Generation of the isomorphic graphs given in Fig. 2 (*c*), (*d*), and (*e*) were avoided.

Although the possibility for generating isomorphic graphs still exists[2], a number of isomorphic graphs can be eliminated with the procedure.

In the above example, a list representation was a convenient way to represent a graph and its symmetry. Other relations of a graph are just as easily represented with lists. An efficient way to perform operations on lists is to use a list processing computer language like LISP.

The author demonstrates an interesting enumeration procedure. I hope that this suggestion to use a graph's symmetry, list representation, and a list processing computer language like LISP will prove to be a usefull addition.

[1]Manager, Mechanical Engineering, Instrument Research and Development, Ames Division Miles Laboratories, 430 S. Beiger Street, Mishawaka, IN 46544

[2]For example, graph 5103 of Figure 6 can be made from either of the graphs shown in Figure 4.

Author's Closure

The author wishes to thank Mr. Pugh for his constructive comments. It is true that we can avoid the enumeration of some isomorphic graphs using the property of symmetry in a graph. However, we must also come up with an algorithm to identify the symmetric vertices in a graph so that the enumeration process can be completely automated by a digital computer. The overall saving in computing time is probably minimal when a quick and efficient method such as the random number technique suggested in this paper can be applied for the identification of isomorphic graphs. Nevertheless, it may be helpful to take the advantage of graph symmetry, list representation, and list processing computer language for the development of a computer-aided mechanism design software.