An Algorithm for the Automatic Generation of Kinematic Chains*

Koichi SUGIMOTO**, ATSUKO HARA**
and Shiyuki SAKAUE**

The computer algorithm for generating all the possible structural patterns of mechanisms with a given number of degrees of freedom, number of general constraints and number of loops has been developed. First, loop patterns composed of links with three or more kinematic pairs, which represent shapes of loops, are determined, and the kinematic graphs are obtained by allocating binary links to some edges of each loop pattern. The number of cases to be considered when allocating binary links to the edges decreases by grouping the edges of loop patterns with the same characteristics and, consequently, the computational time is well reduced.

Key Words: Mechanism, Kinematic Chain, Graph Theory, Structural Synthesis, Number Synthesis

1. Introduction

The design of mechanisms occupies an important part in the development of many types of automatic machines and computer terminals. At present, mechanism design consists mainly of effectively combining existing elements to obtain a mechanism having the necessary function; this method depends greatly on the designers knowledge and experience, and it is not always easy to create a new mechanism this way. To tackle this problem, an algorithm of synthesis of structural patterns has been proposed by which graphs are used to list all the structural patterns according to a determined procedure. Specific methods include an algorithm for generation of the kinematic graphs using the adjacency matrices of the graphs⁽¹⁾, and a method by which a pair of links are added to the basic pattern to generate planar mechanisms⁽²⁾. In addition, it has been proposed that the

patterns obtained in this way are tested for isomorphism of graphs by using the characteristic polynomials of adjacency matrices⁽³⁾⁽⁴⁾. However, with these methods, there are problems such that the automatic generation of patterns by a computer is difficult, and that the algorithm can only be applied to the generation of planar mechanisms. It is difficult to respond to the requirement of generating spatial mechanisms. At present, there is no general and practical method for automatic pattern-generation by computer.

This paper discusses a new algorithm for pattern-generation of kinematic chains by computer. In this method, first, the basic patterns of loops of the chain are generated; then all patterns can be generated by allocating binary links to them. Compared to the conventional method of generating adjacency matrices directly, the frequency of generation of isomorphic patterns is reduced, and, therefore, computational time is greatly reduced.

2. Number Synthesis

When the degrees of freedom F , the number of loops α , and the number of constraints b are given, the number of links and the number of joints can be

* Received 22 nd May, 1987. Paper No. 86-1105 A

** Production Engineering Research Laboratory, Hitachi, Ltd., 292 Yoshida-cho, Totsuka-ku, Yokohama, 244, Japan

determined by the following method. What is meant by the number of constraints here is the number of elements which are unconditionally constrained among the elements of motion of the links which make up the mechanism. In a general spatial mechanism, $b=0$; in a planar mechanism or a spherical mechanism, $b=3$. When F , α and b are given, the number of links N and the number of joints J of the mechanism can be determined by the following formulae.

$$N = F + \alpha(5 - b) + 1 \quad (1)$$

$$J = F + \alpha(6 - b) \quad (2)$$

The joints that are treated here are joints with a single degree of freedom. Joints with multiple degrees of freedom are treated as combinations of joints with a single degree of freedom.

Letting n_m be the number of links connected with m joints, then the following relations are obtained.

$$N = \sum_{m=2}^M n_m \quad (3)$$

$$J = \frac{1}{2} \sum_{m=2}^M m n_m \quad (4)$$

Here, M is the maximum value of m in the mechanism.

In this paper, we assume that a mechanism is not fractionated, that is, a mechanism does not have any parts that can move without affecting the remaining parts, as shown in Fig. 1. The fractionated mechanism can be considered as a set of independent mechanisms. We will not apply the method of synthesis to such fractionated mechanisms. In this case, there is a maximum value which m can take. When finding a solution for n_m , $m=2, 3, \dots, M$, that satisfies the condition given in Eqs. (3) and (4), if $n_M \geq 2$, then by forming loops among two or more links connected with M joints, it is always possible to form a mechanism which does not have any independent loops. Consequently, when it is only possible to form a number of independent loops (the fractionated mechanism), we have:

$$n_M = 1 \quad (5)$$

In a mechanism with $n_M = 1$, let us consider combinations of all the links excluding the link connected with M joints. Considering the link with m_1 joints, as shown in Fig. 2, this can be connected to m_1 joints in

Fig. 1 A chain which can be regarded as two independent mechanisms

the link with M joints. If the linking is done so that this link with m_1 joints shares a joint with a link with m_2 joints, this combination can be connected to $(m_1 + m_2 - 2)$ joints in the link with M joints. Consequently, the combination of all links excepting the link with M joints can be linked to a total of c joints.

$$c = \sum_{m=2}^{M-1} (m-2)n_m + 2 \quad (6)$$

When M is greater than c , it is impossible to connect all of the links such that the chain has only dependent loops. Therefore, in order to exclude the fractionated mechanisms, the following condition must be satisfied:

$$M \leq c \quad (7)$$

However, using Eqs. (1) to (4) and the condition $n_M = 1$, the right-hand side of Eq. (6) becomes $2\alpha + 2 - M$, so condition (7) becomes:

$$M \leq \alpha + 1 \quad (8)$$

That is, the value of M does not become larger than the number of loops plus one.

Using the above conditions, the method of finding the values for n_m when values of N and J are given is as follows.

When a value is given to n_m , $m=2, 3, \dots, M$, the residual numbers of links and joints are defined as:

$$R_n = N - \sum_{m=2}^M n_m \quad (9)$$

$$R_j = 2J - \sum_{m=2}^M m n_m \quad (10)$$

Let d be a number which has n_2 as its first digit, n_3 as its second digit and so on. First, take d to be zero, and add one to the first digit. Initially, the residual numbers R_n and R_j are positive, but after one has been added to d a number of times, either or both of R_n and R_j will become zero. When both become zero at the same time, the values of each digit of d at that time are the solutions for n_m . When at least one of R_n or R_j is zero or negative, then raise a digit; that is, for n_j , for examples set $n_j = n_j + 1$ and, $n_2 = n_3 = \dots = n_{j-1} = 0$. If, after raising a digit, at least one of R_n or R_j is still zero or negative, then raise the next digit. When both R_n and R_j are zero, the values of each digit of d are

Fig. 2 Connection to a link with M joints

the solutions. If the values of R_n and R_j are positive after raising a digit, then add one to the first digit of d again. Repeat this operation until the M th digit of d , that is n_{M+1} , has been raised. By this way, all the solutions which satisfy conditions (3) and (4) for N and J are found. A program in the Pascal language which performs this procedure is shown in Fig. 3.

3. Generation and Classification of Kinematic Chains

3.1 Generation of adjacency matrices

The representation of a kinematic chain by a graph and its adjacency matrix is shown in Fig. 4. In the graph, the links of the mechanism are vertices, and the joints are edges. An adjacency matrix is a matrix which shows the connection relations between vertices of the graph. When two vertices i and j are connected by an edge, the corresponding matrix elements a_{ij} and a_{ji} are one; otherwise, they are zero. An adjacency matrix is a symmetric matrix with all diagonal elements zero. When two vertices are connected by an edge, they are said to be adjacent.

A vertex which shows a link connected with m joints is adjacent to m other vertices. Consequently, the number of elements in a row and column corresponding to a link with m joints is m . Here, m is called the order of this vertex. When the values of n_m , $m=2, 3, \dots, M$ are fixed, so are the orders of all vertices. Let us number the vertices according to their order, starting with the highest. If the order of the i th vertex

```

program combi(input,output);
var J,N,M:integer;
n:array [1..10] of integer;
procedure writeout;
var i:integer;
begin
  for i:=2 to M do write(n[i]:5);
  writeln;
end;

procedure setpair(i,Rj,Rn:integer);
begin
  n[i]:=0;
  while (Rj>=0) and (Rn>=0) do
  begin
    if i=2 then
      if (Rj=0) and (Rn=0)
      then writeout
      else
        setpair(i-1,Rj,Rn);
    n[i]:=n[i]+1;
    Rj:=Rj-i;
    Rn:=Rn-1;
  end;
end;

begin
  read(J,N,M);
  setpair(M,2*J,N);
end.

```

Fig. 3 Program in Pascal language

arranged in this way, is m_i , then the elements of the adjacency matrix must satisfy the relation:

$$a_{i1} + a_{i2} + \dots + a_{i(i-1)} + a_{i(i+1)} + \dots + a_{iN} = m_i \quad (11)$$

where $a_{ij}=1$ or 0.

A matrix whose elements satisfy Eq. (11) is a candidate to be the adjacency matrix. Since the adjacency matrix is symmetrical, $a_{ij}=a_{ji}$. If we find all matrices which satisfy these conditions and then judge which ones can form the required mechanism or have the same structure as a previously determined mechanism, then we have found all structural patterns which correspond to the given values of n_m . However, this method has the defect that many matrices having the same structural patterns appear, and it takes a great deal of time of judge which one can determine a new structural pattern of the mechanism.

3.2 Generation of loop patterns

To reduce the frequency with which isomorphic patterns appear, consider the pattern formed by the vertices which remain after binary links (links connected with two joints) are eliminated, as shown in Fig. 5. This pattern has the same number of loops as the original graph, and the adjacency relations in the loop are the same. In this pattern, it is possible for two vertices to be connected via multiple edges, so it is different from an ordinary graph; let us call this kind of pattern a loop pattern. A loop pattern has its associated adjacency matrix just like an ordinary graph. In a loop pattern, there can be more than one edge connecting two vertices. Consequently, the ele-

Fig. 4 Kinematic graph and its adjacency matrix

Fig. 5 Loop pattern of graph in Fig. 4

edge connecting two vertices. Consequently, the elements a_{ij} of the adjacency matrix do not merely indicate the existence or non-existence of adjacency relations, but show the number of edges connecting vertices i and j , and can take a value of zero or any natural number.

To find a loop pattern, we must allow a_{ij} to be either zero or any natural number in Eq. (11), and find all combinations of solutions for a_{ij} . When this kind of problem is discussed in general terms, it is called the problem of distributing m objects in k locations; in other words, one must solve:

$$B(1) + B(2) + \dots + B(k) = m \quad (12)$$

where each B is an integer such that $B \geq 0$. A sample program for solving Eq. (12) is shown in Fig. 6.

Since the adjacency matrix of a loop pattern is symmetrical, among the elements of the j th row of the adjacency matrix, the values for a_{ij} for $i < j$ are determined by the values of elements from the first to the $(j-1)$ th row; the remaining elements in the j th row can be distributed as in Eq. (12).

In this way, the loop pattern can be determined, but if it is done this way, many isomorphic patterns

```

program combi2(input,output);
var k,m:integer;
    B:array[1..10] of integer;

    procedure writeout;
    var i:integer;
    begin
        for i:=1 to k do write(B[i]:5);
        writeln;
    end;

    procedure setting(i,r);
    begin
        if i=1 then begin
            begin
                B[1]:=r;
                writeout;
            end
        else
            begin
                B[i]:=0;
                while r>=0 do
                    begin
                        if i=2 then
                            begin
                                B[1]:=r;
                                writeout;
                            end
                        else
                            setting(i-1,r);
                    end;
                B[i]:=B[i]+1;
                r:=r-1;
            end;
        end;
    end;

begin
    read(k,m);
    setting(k,m);
end.
```

Fig. 6 Program in pascal language

and improper patterns for mechanisms will be included. Two methods are used together to judge the isomorphic patterns. In the first method, it is judged whether, in the first row of the adjacency matrix, the number of edges which are linked to vertices of the same order are arranged from the highest to the lowest. When the elements of the first row of the adjacency matrix are determined, since the remaining vertices are distinguished only by their order, among vertices of the same order it is possible to arrange them according to the number of edges linked to one vertex, from the highest to the lowest. Consequently, this judgment will reduce the frequency of occurrence of isomorphic patterns. Since the vertices are distinguished by this manipulation of the first row, this judgment cannot be used for the second and subsequent rows.

In the second method, the characteristic polynomial of the adjacency matrix is used, and judgment of isomorphic patterns is carried out by comparing the coefficients of their characteristic polynomials⁽³⁾⁽⁴⁾. This method has not been proven to be theoretically rigorous, but the results obtained to date indicate that there is no problem regarding its practical use⁽⁴⁾, and we have adopted this method in the present work.

An example of an improper kinematic chain is the one shown in Fig. 7, which consists of two separate patterns which are not linked to each other. There is no path that links all of the vertices continuously. Consequently, in a disconnected pattern a generation tree⁽⁷⁾ cannot be created. Therefore, it is attempted to create a generation tree based on the adjacency matrix, and the result can be used to judge the connectedness of the pattern.

3.3 Generation of kinematic graphs

A kinematic graph can be obtained by allocating the remaining binary links to the loop pattern obtained above. However, if all possible graphs that can be formed by allocation of binary links to edges of the loop pattern are obtained, many isomorphic graphs will appear. This is because there are edges which have the same properties with respect to the

Fig. 7 Unconnected graphs

tions among these edges produces equivalent graphs. An example of this is given in Fig. 8, showing what happens when four binary links are allocated to a loop pattern consisting of four ternary vertices. In attempting to obtain one kinematic graph, we have obtained four isomorphic graphs. To prevent these isomorphic graphs from appearing, we must determine which edges in the loop pattern are equivalent with respect to allocation of binary links, and void making allocations that amount to simple interchanges among these edges.

First, let us try allocating one binary link to one of the edges of the loop pattern. Then, allocate a binary link to another edge of the same loop pattern and judge whether or not the two patterns which are obtained by the allocation are isomorphic. If they are isomorphic, then the two edges to which binary links are allocated are judged to be equivalent. This operation can be performed on all of the edges, and the edges are classified into groups of equivalent edges.

Then, binary links are allocated to these separate groups. Select one group for the first allocation of binary links and allocate the predetermined number of binary links to one edge arbitrarily selected in this group. It sometimes happens that this allocation produces a difference between edges that were grouped together initially. Therefore, it is necessary to repeat the grouping. Then, select one from the newly classified groups and allocate the predetermined number of binary links for the second allocation to one of the edges in the group. Repeat this process until allocation of all the binary links yields one kinematic graph. When all possibilities for the distributions of binary links to one pattern, all of the kinematic graphs corresponding to the loop pattern determined by the previously mentioned method are obtained. Repeating this procedure for all distributions of binary links and also all the possible loop patterns, all of the kinematic graphs can be generated.

3.4 Improper kinematic chains

The kinematic graphs obtained in section 3.3

Fig. 8 Isomorphic graphs

include some which are improper because they fall into one of the following two categories.

(1) The graph partially includes structures within the mechanism.

(2) The graph is a combination of two or more independent mechanisms.

Let us now discuss a method for eliminating these improper kinematic chains.

An example of a graph which includes a structure in the mechanism is shown in Fig. 9 for a planar mechanism ($b=3$). In this example, the combination of loop 1 and loop 2 has zero degrees of freedom, and constitutes a structure. When the degree of freedom of the chain in which a combination of several such loops comprises the mechanism, referred to as a sub-closed chain, is zero or less, the mechanism includes a structure. If this sub-closed chain consists of a_s loops, and it has J_s joints, then the degree of freedom F_s of this sub-closed chain is:

$$F_s = J_s - a_s(6 - b) \quad (13)$$

If there exists such a sub-closed chain for which this value is zero or less, then the mechanism should be judged to be improper. In this example of Fig. 9, in the sub-closed chain which consists of a combination of loop 1 and loop 2, a_s is 2 and J_s is 6, so from Eq. (13), F_s is zero. From this, it can be judged that the graph in Fig. 9 is improper.

An example of an improper kinematic chain that violates condition (2) is shown in Fig. 1. Part of it can be eliminated by the condition expressed in Eq. (8), but since this is not perfect, it is necessary to repeat the judgment. Considering the graph which violates condition (2), it has a loop (or loops) which is connected at a fourth-order or higher vertex with another loop or loops, and no edges are shared. The loop (or loops) can move independently with another loop (or loops). Here, we define that two loops are adjacent when they share one or more edges. According to this definition, the adjacency matrix which expresses the adjacency relation of the loops can be determined. From this adjacency matrix, a new graph

Fig. 9 A graph including a structure

Table 1 Number of links and loop patterns

n_2	n_3	n_4	Graph
4	4	0
4	4	0
5	2	1
6	0	2

which expresses the relation of the loops is obtained. In this graph, which is obtained from an improper kinematic chain according to condition (2), a set of graphs which are not mutually connected is obtained. Then, forming the generation tree based on the adjacency matrix of the loops, in an improper kinematic chain according to condition (2), one continuous generation tree cannot be formed. From this judgment, improper kinematic chains according to condition (2) can be eliminated.

4. Examples of Computation

We generated and classified patterns of mechanisms having a single degree of freedom, three constraints and three loops (planar 8-bar linkage). The number of links of this mechanism and the loop patterns are given in Table 1. Four loop patterns were obtained; by allocating the remaining binary links to these loop patterns, sixteen kinematic chains were obtained. These results agree with previously published results⁽⁶⁾.

In the classification of edges of the loop pattern and the judgment of improper kinematic chains discussed in sections 3.3 and 3.4, the classification of edges took by far the longer computational time. Therefore, the classification of edges carried all the way to the end does not always give the shortest computational time. The classification is broken off at a suitable point, and after that, all cases are found and

Table 2 Comparison of computational time

Algorithm (F, b, α)	Allocating binary links	Generating incidence matrices
(1, 2, 2)	0.61 _s	167.37 _s
(2, 2, 2)	0.71	2626.60
(1, 3, 2)	0.53	11.25
(2, 3, 2)	0.55	11.66
(3, 3, 2)	0.69	168.47
(1, 3, 3)	1.84	482.66

the judgment of improper kinematic chains and isomorphic patterns is carried out. This takes far less computational time than carrying the classification all the way to the end. In these examples, we only classified edges the first two times. After that, we did not classify edges, but found all possible cases of allocating binary links, and then performed judgments of improper kinematic chains and isomorphic patterns. The computational times using a Hitachi M-180 computer are given in Table 2. The left-hand column in Table 2 gives the times required to perform the algorithm explained in this paper. The right-hand column gives the times required to directly generate the adjacency matrix of the graph. As can be easily seen from Table 2, use of this algorithm can reduce computational time to 1/2.4 to 1/3 700 of what was formerly required. The more complicated the mechanism, the greater the effectiveness of this algorithm. The programming language used was Pascal.

5. Conclusions

The following results were obtained in this research.

(1) An algorithm was developed for automatically finding the structural patterns of mechanisms having a given number of constraints, degrees of freedom and number of loops using a computer.

(2) The loop pattern of a kinematic graph was defined. A method for allocating binary links to edges of the loop pattern, using the equivalency of those edges, was developed, resulting in a great reduction of the computational time needed to find all of the structural patterns. This method permits the kinematic graph to be generated with computation time reduced to 1/2.4 to 1/3 700 of that required when the adjacency matrix is generated directly.

References

- (1) Raicu, A., Mechanism and Machine Theory, No. 9 (1974), p. 123.
 - (2) Mruthyunjaya, T. S., Mechanism and Machine Theory, Vol. 19, No. 6 (1984), p. 487.
 - (3) Uicker, Jr., J. J. and Raicu, A., Mechanism and Machine Theory, No. 10 (1975), p. 375.
 - (4) Yan, H. S. and Hall, A. S., Trans. ASME J. Mech. Des., Vol. 103, No. 3 (1981), p. 578.
 - (5) Sugimoto, K., J. Robotic Soc. Jpn. Vol. 1, No. 2 (1983), p. 23.
 - (6) Manolescu, N. J., Mechanism and Machine Theory, No. 8 (1973), p. 3.
 - (7) Chachra, V., Ghare, P. M., and Moore, J. M., Application of Graph Theory to Computers, (Translated by Araki and Ioi), p. 57, Kyoritsu Shuppan.
-