Mecánica Computacional Vol. XXIV
A. Larreteguy (Editor)
Buenos Aires, Argentina, Noviembre 2005

TYPE SYNTHESIS OF PLANAR LINKAGE MECHANISMS WITH ROTOIDAL AND PRISMATIC JOINTS

Martín A. Pucheta and Alberto Cardona

Centro Internacional de Métodos Computacionales en Ingeniería (CIMEC)
INTEC (UNL-CONICET),
Güemes 3450 (S3000GLN), Santa Fe, Argentina
e-mail: mpucheta@intec.unl.edu.ar

Key Words: Type Synthesis, Planar Linkage Mechanisms, Graph Theory, Combinatorics.

Abstract. *We present a method to enumerate and codify the solutions of type synthesis of linkage mechanisms with rotoidal and prismatic joints. The essence of mechanism synthesis is to find the mechanism for a given motion. Type Synthesis is the first stage of conceptual design of mechanisms, where the number, type and connectivity of links and joints are determined. It is followed by the Dimensional Synthesis stage, where the link lengths and pivot positions are computed to fulfill a given kinematic task. The latter and the subsequent stages of detailing design are very costly. Therefore, the aim is to propose all "non-isomorphic topologies" without repetitions satisfying structural requirements.*

We use an enumeration of one-degree of freedom topologies developed by Tsai to form an Atlas of kinematic chains. We have developed a method based on the construction of an "initial graph" taking into account prescribed parts (such as fixations, bodies to move, joints, and their interconnections) and the kinematic constraints imposed on them. Then, we use this graph as a pattern to search inside the atlas. This search also involves an isomorphism detection between subgraphs occurrences inside a kinematic chain of the atlas. We develop a classification of the occurrences by the Degree Code of the edge-induced subgraph produced by the pattern edges. After selection of a non-isomorphic topology, there follows a step of specialization where joint types are assigned. We also developed a method to enumerate all possibilities of joint assignments in a non-isomorphic way for a maximum number of prismatic joints given by the user.

The method is illustrated with examples for typical aeronautical test problems. The program was written in C++ *language under the OOFELIE environment.*

1 INTRODUCTION

A *kinematic chain* is a chain with mobility, without any fixed link. In the bibliography it is frequently referred to as Basic Kinematic Chain (BKC). When one or more links of a BKC are fixed, it becomes either a *mechanism* if at least two links retain mobility, or a *structure* if it does not have mobility. A mechanism is a mechanical device that has the purpose of transferring motion and/or force from a source to an output body.

Type synthesis consists in selecting the type of mechanism and the type and number of its component parts: it could be linkage, gear, cam, belt, etc. or a combination thereof. In this work we are limited to planar mechanisms of the linkage type. A *linkage* consists of links (or bars), usually considered rigid, which are connected by joints such as pins (or revolutes) or prismatics, to form open or closed chains. Linkages can be designed to perform complex tasks, such as nonlinear motion and force transmission[1].

Since the early eighties, there has been an increasing interest for the integration of techniques for *computer-aided synthesis of linkages* into CAD-CAE simulation tools[2], particularly addressed to the analysis and optimization of mechanisms. However, the problem of synthesizing a mechanism for a given kinematic objective has been less treated in the literature. The essence of this problem is to find the mechanism for a given motion or *task*. In other words, it deals with the systematic design of mechanisms for a required performance (Figure 1).

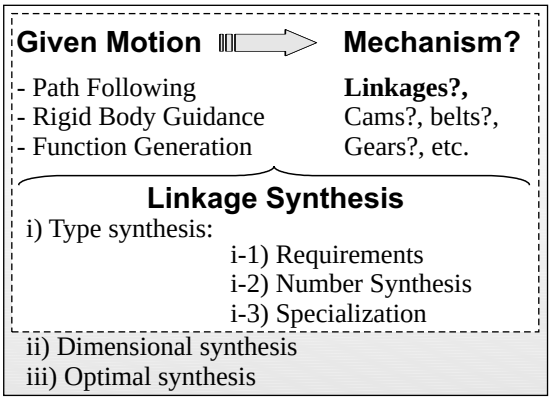


Figure 1: Methodology scope (in dashed)

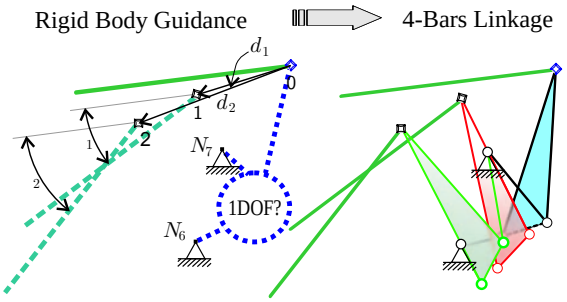


Figure 2: Example of linkage synthesis

There are three customary *tasks* for kinematic synthesis: path following (PF), rigid-body guidance (RBG), and function generation (FG). In the case of path following, the purpose of the synthesis is to determine the dimensions of the elements so that one or more points of the mechanism move through a series of preset positions. In that of rigid-body guidance, the positions and orientations of one or more elements are prescribed (see example in Figure 2). Function generation intends to coordinate motion between input and output links of the mechanism.

Mechanism synthesis methods are usually decomposed into two phases: (i) *Type Synthesis*, where the number, type and connectivity of links and joints are determined[3–5], and (ii) *Dimensional synthesis*, where the proper dimensions of parts are computed[1,6–8]. In the linkages case,

the dimensional synthesis consists in computing the links lengths and pivots positions. This task is also related to that of optimization of the mechanism, aspect which has been dealt with by various authors[9–11]. The dimensional synthesis phase and all the subsequent stages of detailing design are very costly, so it is essential to have good topology alternatives as output of the type synthesis stage in order to avoid repeated simulations and also not to leave some alternatives without being explored.

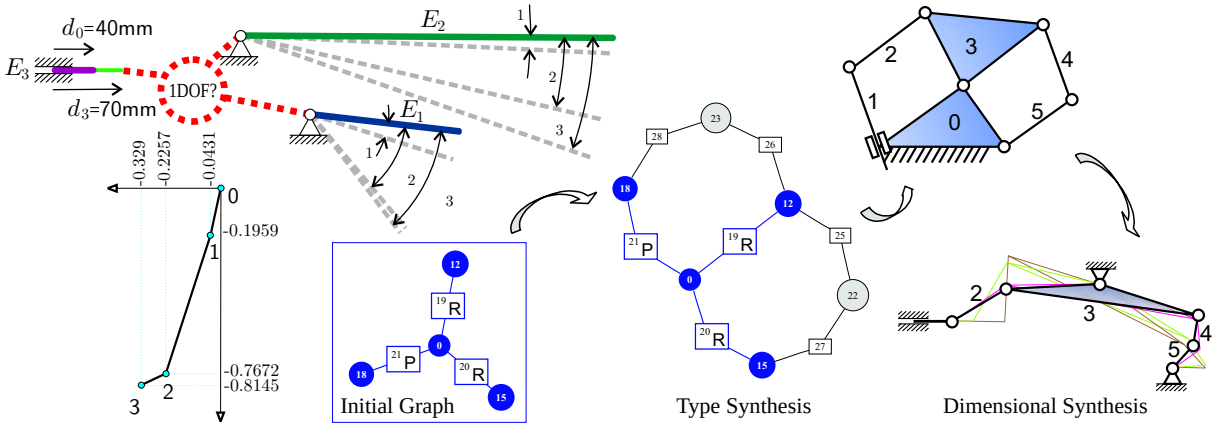


Figure 3: Synthesis of a combined task

Algorithmic solutions to type synthesis problems were proposed since the mid-sixties. Graph Theory[1] has been used for analysis and enumeration of rigid linkages by Freudenstein and Dobrjanskyj, Crossley, Manolescu[3]; and more recently Tsai[5] who developed the atlas of linkages that we adopt. The graph $G(E, V)$ of a kinematic chain is obtained by representing each link by a vertex v_i and each kinematic pair by an edge e_i connecting the corresponding vertices. By convention, the *ground* is the vertex v_0 labeled as zero.

The number of solutions obtained from the Type Synthesis phase could be finite but numerous; depending on the problem, we may have even hundreds or thousands of solutions. The choice of a suitable mechanism for the desired purpose may be done by "visual inspection" on an atlas of linkages but Graph Theory may be conveniently applied to do this search automatically. In the past, the search in atlases was often carried out based on the intuition of experienced designers, but such a procedure may easily lead to neglecting possible solutions.

The automatic tool developed under the OOFELIE[14,15] environment is a unified and systematic approach to an "oriented-task type synthesis of linkages with lower pairs" that puts at the user's disposal a list of all non-isomorphic mechanism alternatives potentially suited to develop a required task. The solutions are sorted and codified in an increasing complexity order. The algorithms are also generalizable to more complex cam-linkage and geared-linkage mechanisms with higher pairs. They also serve as an introduction to the field of enumeration of flexible linkages.

[1]Some background from Graph Theory can be found in the books of Harary[12] or West[13].

The paper is organized as follows: the process of Type Synthesis is reviewed in Section 2. The graph representation of kinematic problems is shown in Section 3. The enumeration of kinematic chains is reviewed in Section 4. The proposed Number Synthesis algorithm is developed in Section 5. The proposed algorithms for prismatic joints assignment is developed in Section 6. The topics are developed with detailed examples.

2 TYPE SYNTHESIS PROCESS

Olson[3], Erdman and Sandor[10], Tsai[5], and Yan and Hwang[4], gave guidelines for a type synthesis strategy. The process that we adopt consists in:

I Problem requirements: From the required design specifications, the designer selects the *structural characteristics* such as : *prescribed parts* defined by fixations, bodies to move, joints, and their interconnections, and the *kinematic constraints* imposed on them. Then, the user sets the required number of degrees of freedom of the mechanism, the allowed joint types and the maximum number of joints desired for each type. The prescribed parts and kinematic constraints are defined using a CAD interface. From the interpretation of this drawing a so called *initial graph* is automatically constructed to define the problem, as we explain later.

II Number Synthesis: The second step is to enumerate the atlas of the kinematic chains with the required numbers of links and joints. Methods to obtain an *atlas* of non-degenerate and non-isomorphic planar kinematic chains with simple joints are available from works of W. Hwang and Y. Hwang[16], Hsieh[17], and Tsai[5]. We use them making a subgraph search and codifying the occurrences of the initial graph. We dispose the atlas as a list of integers codifying each BKC by its Degree Code (DC). They are sorted by increasing complexity order. The identification of an *objective link* constraints and reduces in cardinality the enumeration of solutions. Depending on the task, it must be at a given distance from the ground.

III Specialization: The third step is to specialize each kinematic chain by assigning the type of links and joints (Yan and Hwang[4]). The user's constraints limit the number of permutations needed to obtain all possible configurations of mechanisms in a non-isomorphic way. The desired number of joints of each joint type is used as an input constraint. The types of joints pertaining to the initial graph are respected and not changed.

3 GRAPH REPRESENTATION OF KINEMATIC PROBLEMS

Starting from functional requirements, the designer selects the *structural characteristics* to draw the existing parts like a *skeleton diagram*. Many dimensions of this diagram are unknown (to be synthesized), but some of them can be imposed. On the imposed parts the user gives the motion constraints to define the task. To initiate the Number Synthesis stage, the drawing is converted into an abstract structural representation of prescribed parts that we call "initial graph" (Subsection 3.2). This graph makes it possible to systematically explore potential linkages -using combinatorial theory- inside an atlas of kinematic chains.

3.1 Mechanism class representation

A mechanism is represented internally in a multiple-set of data $\mathcal{M} = \{\boldsymbol{N}, \boldsymbol{F}, \boldsymbol{E}\}$, consisting of nodes, fixations and elements with attributes like positions, types of element, connectivities, etc. This representation is entered to the program interactively from a CAD interface which is common to other finite element analysis programs. The user may also interact with the program in a special language developed for communication.

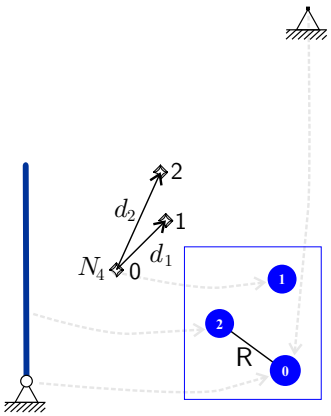


Figure 4: Path following.

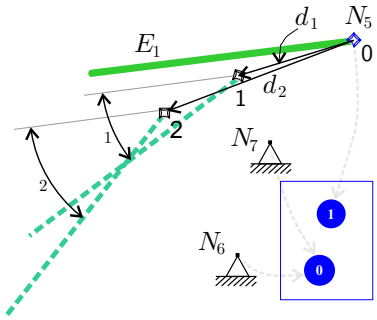


Figure 5: Rigid-body guidance.

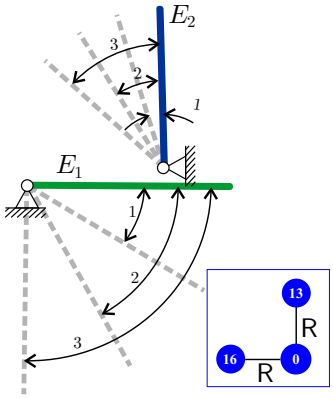


Figure 6: Function generation.

In addition, the user should specify requirements for the synthesis task. We may distinguish three cases:

Path Following. To define a trajectory, we give a set of node displacements

$$\mathcal{D} = \{N_{ID}, t, (d_x, d_y); ...; \},$$

where ID is the node identifier, t is the number of passing points in the sequence of precise positions, and displacements are expressed in relative coordinates from initial node position. For instance, if two displacements are desired on node N_4, we declare three triplets $\mathcal{D} = \{N_4, 0, (0, 0); N_4, 1, d_1; N_4, 2, d_2\}$ (Figure 4).

Rigid-Body Guidance. Here, displacements and also orientations are defined for an isolated node. We declare the displacements $\mathcal{D} = \{N_5, 0, (0, 0); N_5, 1, d_1; N_5, 2, d_2\}$ on node N_5, and also rotations with the triplets $\mathcal{L} = \{E_1, 0, 0; E_1, 1, \alpha_1; E_1, 2, \alpha_2\}$ for rigid-body E_1 (Figure 5).

Function Generation. Now, two (or more) sequences of displacements or orientations are specified for two (or more) grounded rigid-bodies. For instance, a law $\boldsymbol{\beta} = f(\boldsymbol{\alpha})$ is given for two bodies hinged to ground in Figure 6. We have defined the sets $\mathcal{L} = \{E_1, 0, 0; E_1, 1, \alpha_1; E_1, 2, \alpha_2; E_1, 3, \alpha_3\}$ and $\mathcal{L} = \{E_2, 0, 0; E_2, 1, \beta_1; E_2, 2, \beta_2; E_2, 3, \beta_3\}$.

Motorized prismatic joints can also be present in a mechanism, and in this case input displacements may be given for them. For instance, in Figure 3, the initial and last positions are prescribed for the prismatic joint E_3 in the form $\mathcal{J} = \{E_3, 0, d_0; E_3, 3, d_3\}$. The intermediate inputs are unknown and are computed by the dimensional synthesis program. This problem is a *double function generation*, and data is completed by imposing angular displacements at link E_1: $\mathcal{L} = \{E_1, 0, 0; E_1, 1, \beta_1; E_1, 2, \beta_2; E_1, 3, \beta_3\}$ and at link E_2: $\mathcal{L} = \{E_2, 0, 0; E_2, 1, \alpha_1; E_2, 2, \alpha_2\ E_2, 3, \alpha_3\}$.

3.2 Initial graph

The initial graph represents the initial parts. The initial graph cannot have more than two connected components. If more components appear, the problem is subdivided into successive synthesis tasks. For instance, a wing flap/tab coordination problem can be divided into a "flap guidance" problem and a subsequent "tab guidance" problem[18].

Tasks can be distinguished by the number of graph components in the initial graph associated with the problem (see, e.g. Figures 3 to 6). For Path Following (PF) and Rigid-Body Guidance (RBG) tasks, the initial graph has two graph components. One component includes the ground and the other component has always one isolated vertex which is taken as coupler link. This isolated vertex will have at least one physical node with prescribed movements. The prescribed movements could be a set of translations $\mathcal{D}$ for PF , or sets of translations $\mathcal{D}$ and rotations $\mathcal{L}$ for RBG. Since the name "coupler or floating link" comes from the studies of the traditional four-bar mechanism, we will rename it to "*objective vertex*".

For a Function Generation (FG) task a link with imposed joint movements $\mathcal{J}$ is considered as "driver link", so that part of the problem is to detect how many choices of output links can be selected. On the other hand, for PF and RBG problems the input link (driver) cannot be prescribed and the resulting options for input links could change the behavior of the mechanism conducting the driven links through different kinematic circuits, and therefore different workspaces. In case the *timing* is not prescribed, the motorization set of movements $\mathcal{J}$ must be computed in later dimensional synthesis stages.

In this way, the Input/Output (I/O) relationship desired for the synthesized mechanism is implicitly entered by the user when he specifies the sets $\mathcal{D}$, $\mathcal{J}$, and $\mathcal{L}$. The problem of synthesis of mechanisms is to find a mechanism capable of developing a prescribed I/O relationship minimizing the error between the desired and the generated movements[2].

Starting from the initial mechanism definition, $\mathcal{M}_{ini}$, we build the associated *initial graph* G_{ini} following these simple rules:

Vertices: Free bodies with imposed movements will be isolated vertices of the initial graph. These may result in an *objective vertex* that appear, for instance, in the PF problem (Figure 4) or in the RBG problem (Figure 5).

[2]The most primitive level of conceptual design synthesize mechanisms from the nature of the I/O movements selecting all combination of mechanisms, not only restricted to the linkage type. At respect, there are available methodologies developed by Moon and Kota[19], Chiou and Kota[20], and also Wang and Yan[21].

The remaining bodies, connected through joints, will be connected vertices of the graph. Conventionally, the ground link will be the vertex zero. Depending on the number of grounded bodies, this vertex may be binary, ternary, etc.

Edges: All joints will be edges of the initial graph connecting the previous defined vertices. All joints are assumed to be binary. Isolated joints are not allowed, so they should be at least connected to one vertex.

4 ENUMERATION OF KINEMATIC CHAINS

There are two main categories within the Graph Theory strategies for enumeration of KC's:

1. Those that enumerate the feasible topologies using the Grüebler equation, taking into account the number of degrees of freedoms that each joint/element constraints. The use of "contracted graphs" decreases dramatically the complexity for isomorphism testing because KCs with equal contracted graphs share many characteristics like the number of links (thus the adjacency matrices have the same order), number of joints, number of loops and the degrees of non-binary vertices. Hwang[16] gave a computational way to deal with twelve links BKCs. Hsieh[17] developed enumerations using dual graph and dual contracted graphs. Recently, Tsai[5] provided atlases of bar-linkages, planetary gear trains, and robotic mechanisms.

2. The enumeration of topologies by addition of "Assur groups" to the basic four-bars mechanism preserving the mobility criterion (Equation 4 in the next subsection). An individual Assur group is an open chain which satisfies $3n - 2j = 0$, where n is the number of rigid members of the chain and j is the number of joints.

From any of these resulting enumerations considering only revolute pairs we can then apply an equivalence criteria (Franke) to form mechanisms with more complicated pairs, e.g. higher-pairs. For instance, this can be seen when replacing a revolute-dyad by a cam or a gear pair, or even a slider or an hydraulic cylinder[22]. Using Graph Theory, all mechanisms can be generalized from an enumeration of revolute-mechanisms. A methodology for generalization issues was developed by Wang and Yan[21].

4.1 Basic formulas

Let F be the number of degrees of freedom of the desired mechanism, λ the number of degrees of freedom of the space in which the mechanism is intended to function (using $\lambda = 3$ for planar/spherical motion and $\lambda = 6$ for spatial), and c_i the degrees of constraint on relative motion imposed by joint i. Mechanism structures can be enumerated satisfying the Grüebler equation:

$$F = \lambda(n-1) - \sum_{i=1}^{j} c_i \tag{1}$$

where n is the number of links, and j the number of joints. The constraints imposed by a joint and the degree of relative motion f_i permitted by the joint i are related by:

$$c_i = \lambda - f_i \tag{2}$$

So, the substitution of (2) into (1) leads to the Grübler or Kutzbach criterion:

$$F = \lambda(n - j - 1) + \sum_{i=1}^{j} f_i \tag{3}$$

Since rotoidal and prismatic joints permit only one d.o.f., the equation for a planar ($\lambda = 3$) mechanism with j joints is:

$$F = 3(n - j - 1) + \sum_{i=1}^{j} 1 = 3(n - j - 1) + j = 3(n - 1) - 2j \tag{4}$$

The number of independent loops L could be computed from graph theory, as:

$$L = j - n + 1 \tag{5}$$

Conversely, combining (5) and (4) yields

$$F = \lambda \underbrace{(n - j - 1)}_{-L} + \sum_{i=1}^{j} f_i \quad \Rightarrow \quad \sum_{i=1}^{j} f_i = F + \lambda L \tag{6}$$

also known as the *loop mobility criterion*, useful for determining the number of joint degrees of freedom needed for a KC to possess a given number of degrees of freedom.

A simple running over the link and joint numbers n and j from 1 to 30 satisfying $F = 1$ in the Equation 4, gives the results shown in Table 1 where NaM means "not a mechanism" and NE means "not enumerated".

Methods given by Hwang[16], Hsieh[17] or Tsai[5] applying the first mentioned strategy allow us to find the first nineteen BKC's from the 7111 that they found. For one-dof mechanisms they established that there are one kinematic chain with one loop (the well known four-bar mechanism); two KCs with six bars (6,7), known as Watt and Stephenson chains; sixteen KCs with eight bars (8,10); 230 KCs with ten bars (10,13); 6862 KCs with twelve bars (12,16).

These KCs were found using graph theory where the feasible graphs have the following characteristics: (i) the minimal vertex degree is 2 ($d(v_i) \geq 2$); (ii) all graphs have no articulation points or bridges; (iii) partially locked chains/subchains, and non planar graphs were excluded. Actually, the practical interest goes from 1 up to 5 loops mechanisms. In the fifth column of Table 1 the pictures of each graph with its corresponding *Level* and Degree Code are shown. As it will be explained next the Degree Code allows to obtain the graph.

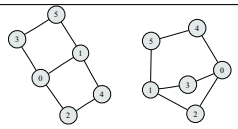
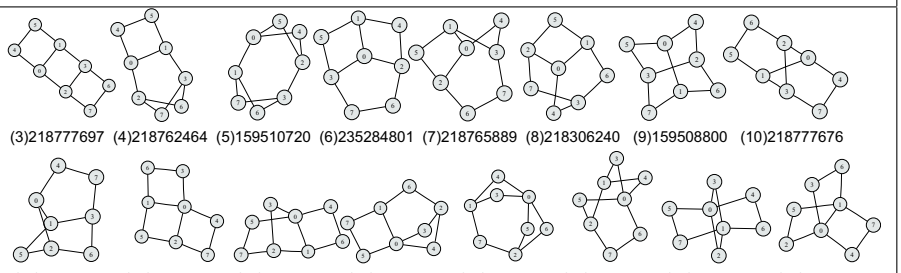
n	j	L	BKC's	Kinematic Graph (Level) Degree Code
2	1	0	NaM	-
4	4	1	1	 (0)51
6	7	2	2	 (1)28882 (2)15169
8	10	3	16	 (3)218777697 (4)218762464 (5)159510720 (6)235284801 (7)218765889 (8)218306240 (9)159508800 (10)218777676 (11)218765512 (12)251860104 (13)194069522 (14)193062914 (15)63740928 (16)64754689 (17)127500306 (18)251891848
10	13	4	230	-
12	16	5	6862	-
14	19	6	NE	-
16	22	7	NE	-
18	25	8	NE	-
20	28	9	NE	-

Table 1: Pairs (n, j) of the number of links and joints for linkages with one degree of freedom, and the information relative to the number of independent loops L, and the codified found solutions.

4.2 Tools for Topology analysis

4.2.1 Degree code

In order to represent graphs and to detect isomorphism of graphs, we used the so-called *Degree Code* developed by Tang and Liu[23]. It is built from the adjacency matrix of the graph, which is defined next.

> Adjacency matrix $A(G)$: n-by-n matrix in which entry a_{ij} is the number of edges in G with endpoints $\{v_i, v_j\}$ and $a_{ii} = 0$. Note that permutations of labels change the adjacency matrix $A(G)$, in other words, it is label order dependent.
>
> MAXcode: Maximum integer that results from converting to decimal the binary obtained by concatenating the upper triangular elements of the adjacency matrix row by row, excluding diagonal elements, among all possible vertices label orders. For n-vertices, $n!$ permutations are needed. Graphs sharing the same MAXcode are isomorphic.

Degree code (DC): It is computed following the same procedure that the MAXcode, but permutations are taken by subgroups of vertices with the same degree, so that the required number of permutations is reduced.

The degree code is *unique* (two graphs with the same DC are isomorphic: $DC(G_1) = DC(G_2) \Leftrightarrow G_1 \cong G_2$) and *decodable* (given the DC, we may build the graph). Then, as it is shown in Table 1, the entire atlas of one degree-of-freedom mechanisms may be very efficiently stored as a sorted list of integers. For instance, a simple four bar kinematic chain labeled as we see in the first row of Table 1 can be characterized by the integer 51:

$$A_1 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, DC(A_1) = ([110][01][1])_2 = 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^4 + 1 \times 2^5$$
$$= 1 + 2 + 16 + 32 = (51)_{10}.$$

Tang and Liu also presented a method to take into account colored edges. In our case each color represents a joint type and the type is represented by a number: 1 if the joint is rotoidal and 2 for prismatic. Calling b the number of joint types plus one, and taking it as the base for the numerical system, the Degree Code in base three ($b = 2 + 1$) of the following matrix

$$A_2 = \begin{bmatrix} 0 & 2 & 2 & 0 \\ 2 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \text{is } DC_3(A_2) = ([220][01][1])_3 = 1 \times 3^0 + 1 \times 3^1 + 2 \times 3^4 + 2 \times 3^5$$
$$= 1 + 3 + 162 + 486 = (571)_{10}.$$

In order to consider colored diagonal elements, we extend the idea of Tang and Liu to define a *Diagonally Extended Degree Code DC_b^d in base b*. For example, let A_3 be a matrix which represent a topology with different types of links and joints (explained later in Subsection 6.1); its codification may be computed as follows:

$$A_3 = \begin{bmatrix} 2 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \end{bmatrix}, DC_3^d(A_3) = ([2011][112][00][1])_3 = 1 \times 3^0 + 2 \times 3^3 + 1 \times 3^4 +$$
$$+ 1 \times 3^5 + 1 \times 3^6 + 1 \times 3^7 + 2 \times 3^9 = (21503)_{10}.$$

Note that the needed base b is the maximum integer appearing in any entry of A_3 plus one even if they are representing links (diagonal) or joints (outer diagonal entries). See Appendixes B and C for programming details.

4.2.2 Classes of similar vertices

A *permutation* is said to be *automorphic* if it maps the adjacency matrix of a labelled kinematic chain onto itself (Figure 7). The automorphic permutations of a kinematic chain form a group, the so called *group of automorphisms*. Two vertices v_i, v_j are similar if there exists a permutation p of the group of automorphisms which transforms v_i into v_j.

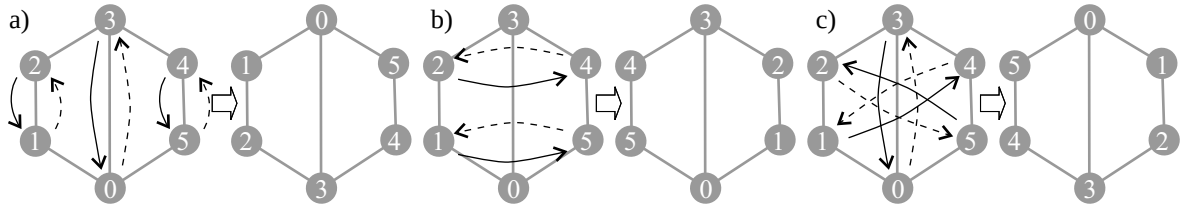


Figure 7: Automorphic permutations of the Watt kinematic chain (the identity permutation is not drawn).

Yan[4] proposed an algorithm based on the analysis of the group of automorphisms to detect the classes of similar vertices. In this work, we use a method to find these classes based on the Diagonally Extended Degree Code DC_b^d in base 2. Given a topology taken from the atlas, the procedure is to assign one non-null element on the diagonal of its adjacency matrix, $a_{ii} = 1$, and compute the DC_2^d. Then, vertices of the same class are those which have the same codes (Figure 8).

For instance, a four-bars kinematic chain has only one class of vertices. If we put a 1 on the diagonal, precisely on the position a_{00}, the diagonally extended degree code is 906, so the vertex v_0 has class 906.

$$A_0 = \begin{bmatrix} \text{'1'} & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, DC_2^d(A_0) = ([1110][001][01][0])_2$$

$$= 1 \times 2^1 + 1 \times 2^3 + 1 \times 2^7 + 1 \times 2^8 + 1 \times 2^9$$

$$= 2 + 8 + 128 + 256 + 512 = (906)_{10}.$$

Due to the symmetries of the four-bars KC, the same class is obtained for the other positions a_{ii}, $i = 1, 2, 3$.

A Watt six-bars KC has two classes, the 1714498, and 1969288 (Figure 8). A Stephenson six-bars KC has three classes: 1520642, 1707456, and 18414743.

Once a vertex is chosen to be the ground, a KC is converted into a mechanism. By changing the vertex selected to be the ground we get different mechanisms for the same KC, an effect which is called *Linkage Inversion*. Classes of similar vertices allow us to identify all non-isomorphic vertex assignments.

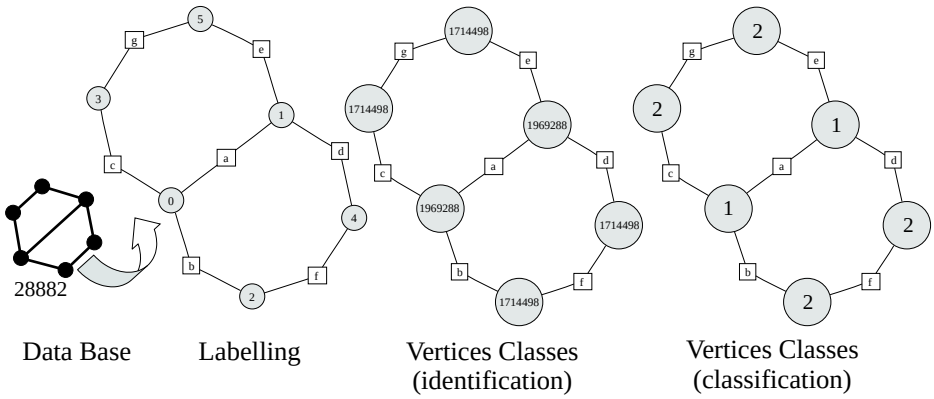


Figure 8: Watt topology and its classes of similar vertices.

For instance, choosing as ground a vertex of class 1714498 in a Watt KC results a Watt I mechanism. Instead, if a vertex of class 1969288 is chosen, it results a Watt II mechanism. Another example could be a Stephenson KC. Choosing vertices of class 1520642, 1841474, and 1707456, results in the Stephenson I, II, and III mechanisms respectively.

4.2.3 Distance from the Objective Vertex

The objective vertex corresponding to the coupler point, i.e. the vertex containing the node which develops the prescribed task, is chosen counting a defined distance from the ground. This distance is defined as the minimal number of vertices going through a path beginning from the objective vertex to the ground.

For FG problems this constraint is not computed because there is not objective vertex. For PF and RBG tasks, the minimum distance from the objective vertex to the ground should be 2 while the maximum is either fixed by the user or taken to be equal to the number of passing points specified in the task minus one. For instance, if three passing points are specified for a PF task, the vertex is chosen at a distance of two from the ground. Of course, depending on the topology, there can be more than one options for the objective vertex that verify this requirement (although some of them can be isomorphic).

The subgraph search proposed in the next section assigns automatically the objective vertex.

4.2.4 Edge-Induced Subgraph

An *Edge-Induced Subgraph* of the graph G is obtained by deleting a subset of edges E_{ini} from the set of edges E pertaining to $G(E, V)$. It can be simply obtained deleting edges (ones) from the adjacency matrix of G. The Degree Code of the resulting graph $S = G - E_{ini}$ is used to classify the topology of the parts added to an initial graph to complete a KC.

5 SUBGRAPH SEARCHING

The initial graph represents the initial situation. In order to get a mechanism that matches this initial situation, the initial graph should be a subgraph of any valid mechanism of the atlas of mechanisms. Then, the first step of computation consists in looking for the simplest mechanism in the atlas for which the initial graph is a subgraph.

$$G_{ini} \subseteq G_A \tag{7}$$

with G_A a graph from the atlas.

The search begins in the lowest level of complexity for the number of degrees of freedom required for the mechanism. In this way, we try to *minimize* the number of links in the solution. By default, the number of degrees of freedom is one.

Let $L = \{l_0, l_1, ..., l_{ini-1}\}$ be the labels of the initial graph G_{ini}, and $\mathcal{V}$ the function which applies the set L to the set of unlabeled vertices $V = \{v_0, v_1, ..., v_{ini-1}\}$, so that $\mathcal{V}(v_i) = l_i$.

The algorithm for selection of a suitable mechanism is the following:

S0. Initialize search level $A = -1$.

S1. Increase level index A and take a graph G_A from the atlas, with n_A vertices. If $n_{ini} \leq n_A$, continue to **S2**; else, repeat **S1**.

S2. For each permutation G_A^p of G_A, take a subgraph H_A of the permuted graph G_A^p. The vertices of H_A are composed by the fist n_{ini} vertices of the permuted list of vertices of G_A^p.

$$p(V_A) = p(\{v_0, v_1, ..., v_{n_A}\}) = \{\underbrace{v_0^p, v_1^p, ..., v_{ini-1}^p}_{V(H_A)}, ..., v_{n_A}^p\}$$

Label H_A using $\mathcal{V}$. There are $\binom{n_A}{n_{ini}} n_{ini}!$ permutations p to be explored following the lexicographic order of vertex labels. If all the n_{ini}-permutations have been tested in the explored level, return to **S1**.

S3. If H_A is isomorphic to G_{ini}, $H_A \cong G_{ini}$, then $G_{ini} \subseteq G_A^p$. Check if connections of the graph using the labelled vertices are the same in H_A and G_{ini}. If it is true, label the remaining vertices of G_A^p with labels starting from $\max(l_i) + 1$, $i = 0, 1, ..., n_{ini}$; set $G_a \leftarrow G_A^p$ and exit. Else, return to **S2** and choose a new subgraph H_A in lexicographic order.

At the end, mechanism $\mathcal{M}_a$ with graph G_a will be the simplest admissible topology. The process may be repeated resulting in a list of mechanisms with admissible topologies, $\mathcal{M}_a$, $\mathcal{M}_b$, $\mathcal{M}_c$, etc. These mechanisms inherit synthesis data definitions ($\mathcal{D}$, $\mathcal{J}$, $\mathcal{L}$) from $\mathcal{M}_{ini}$, useful to compute the missing data (i.e. vertices representing new links have unknown node positions) at later dimensional synthesis stages.

In this way, we performed the *number synthesis* process, that is, we determined the topology (degrees of freedom, number of links, number of joints and their interconnections) of a suitable mechanism that could answer the requirements.

In order to distinguish among topologies G_a, G_b, G_c, etc., we develop a vectorial codification. Each new generated topology is assigned a new vector, which is compared entry-by-entry with those already stored.

Let us define the *edge-induced subgraph* S as the graph obtained by deleting from graph G_A the subset of edges pertaining to subgraph G_{ini}. *Isomorphic occurrences* may be detected by requiring not only that $G_{ini} \subseteq G_A^p$, but also the DC of S, the distance constraints, and the classes of its vertices, be different from previous answers. The complete vector for identification and comparison between the yet stored topologies in **S3** consist then of the following four fields:

$$[A, DC(S), \boldsymbol{d}, \boldsymbol{C}]$$

where,

A: Integer representing the Level in the atlas (position of the stored Degree Codes of BKCs),

$DC(S)$: Degree code of the edge-induced subgraph S,

$\boldsymbol{d}$: Distances from the objective vertex v_{obj} to the remaining vertices of the initial graph in G_A^p given by $\boldsymbol{d} = (d(v_{obj}, v_0), d(v_{obj}, v_1), d(v_{obj}, v_2), ..., d(v_{obj}, v_{n_{ini}-1}))$. Only the distance from the objective vertex to the ground is constrained by user's limits $d_{min} \leq d(v_{obj}, v_0) \leq d_{max}$. The remaining are used for classification ends and could take any value.

$\boldsymbol{C}$: Vector of classes $C(v_i)$ of each vertex of the initial graph inside the selected graph G_A^p

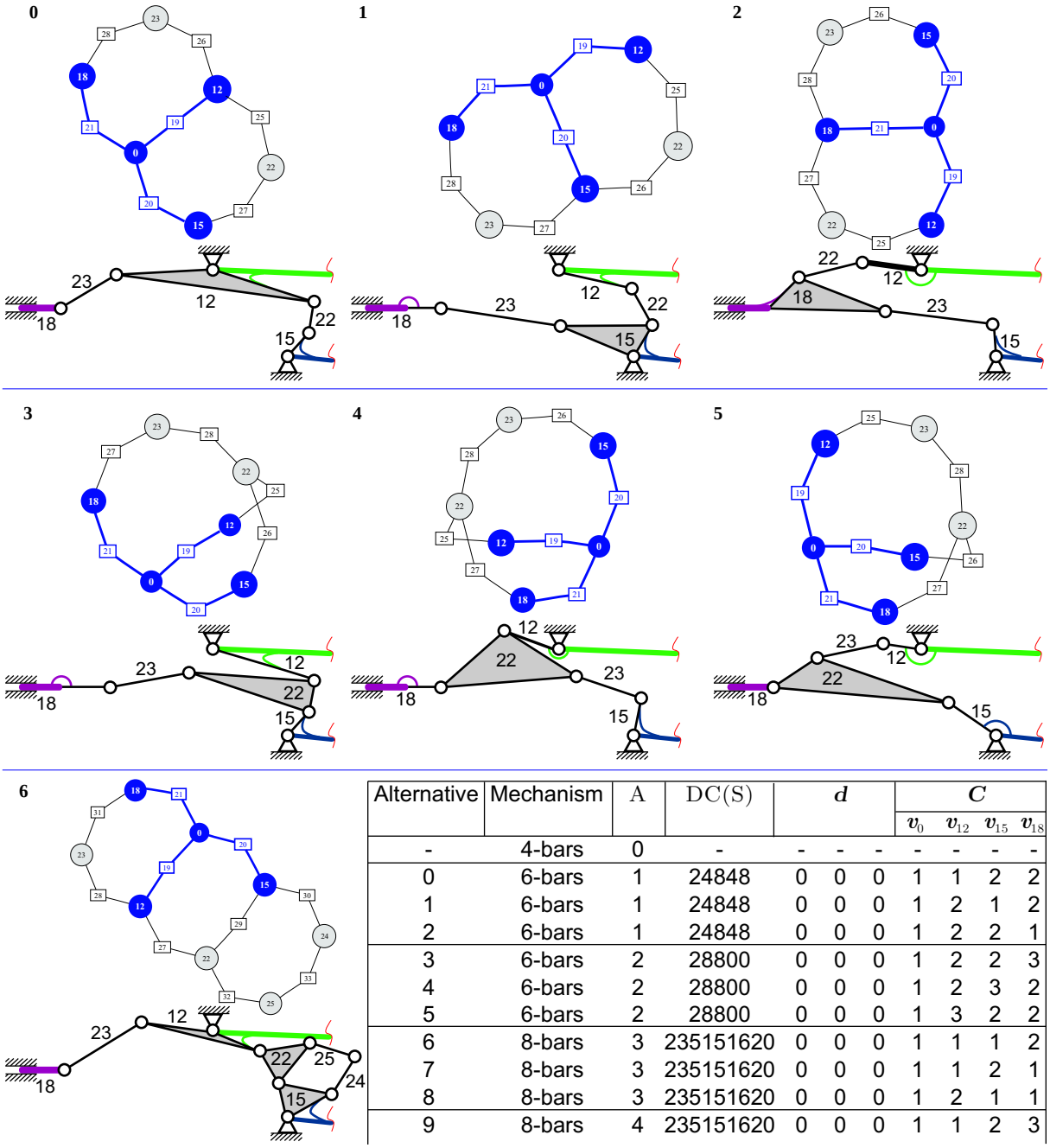
$$\boldsymbol{C} = (C(v_0), C(v_1), C(v_2), ..., C(v_{n_{ini}-1})).$$

Their meaning is illustrated next with practical examples.

Example 1 Double Function Generation

The problem shown in Figure 3-left schematizes the coordination between two flaps of a turbine engine and the horizontal movement of an hydraulic cylinder. There were 191 solutions found in the atlas using the present method. The first seven solutions are shown in Figure 9. For clarity, an schematic sketch is drawn below each solution.

Note that the distance vector $\boldsymbol{d}$ was not computed because there is not objective vertex. From solutions 0, 1, and 2, we see that the vertices labeled with 12 (primary flap), 15 (secondary flap), and 18 (cylinder), have all possible "class vertex" combinations at the different topologies. On the other hand, the ground only takes the class 2 (Watt-II) because its degree is prescribed to $d(v_0) \geq 3$. The same observation is valid for the solutions 3, 4, and 5 where only Stephenson-III mechanisms are obtained. □



Alternative	Mechanism	A	DC(S)	d			C			
							v_0	v_{12}	v_{15}	v_{18}
-	4-bars	0	-	-	-	-	-	-	-	-
0	6-bars	1	24848	0	0	0	1	1	2	2
1	6-bars	1	24848	0	0	0	1	2	1	2
2	6-bars	1	24848	0	0	0	1	2	2	1
3	6-bars	2	28800	0	0	0	1	2	2	3
4	6-bars	2	28800	0	0	0	1	2	3	2
5	6-bars	2	28800	0	0	0	1	3	2	2
6	8-bars	3	235151620	0	0	0	1	1	1	2
7	8-bars	3	235151620	0	0	0	1	1	2	1
8	8-bars	3	235151620	0	0	0	1	2	1	1
9	8-bars	4	235151620	0	0	0	1	1	2	3

Figure 9: First seven non-isomorphic occurrences of an initial graph (pattern) inside the atlas.

Example 2 Path Following

For the problem shown in Figure 4, there are 480 alternatives inside the atlas of 19 BKCs for an allowed distance from the objective vertex to ground, constrained to take the values 2 or 3. The

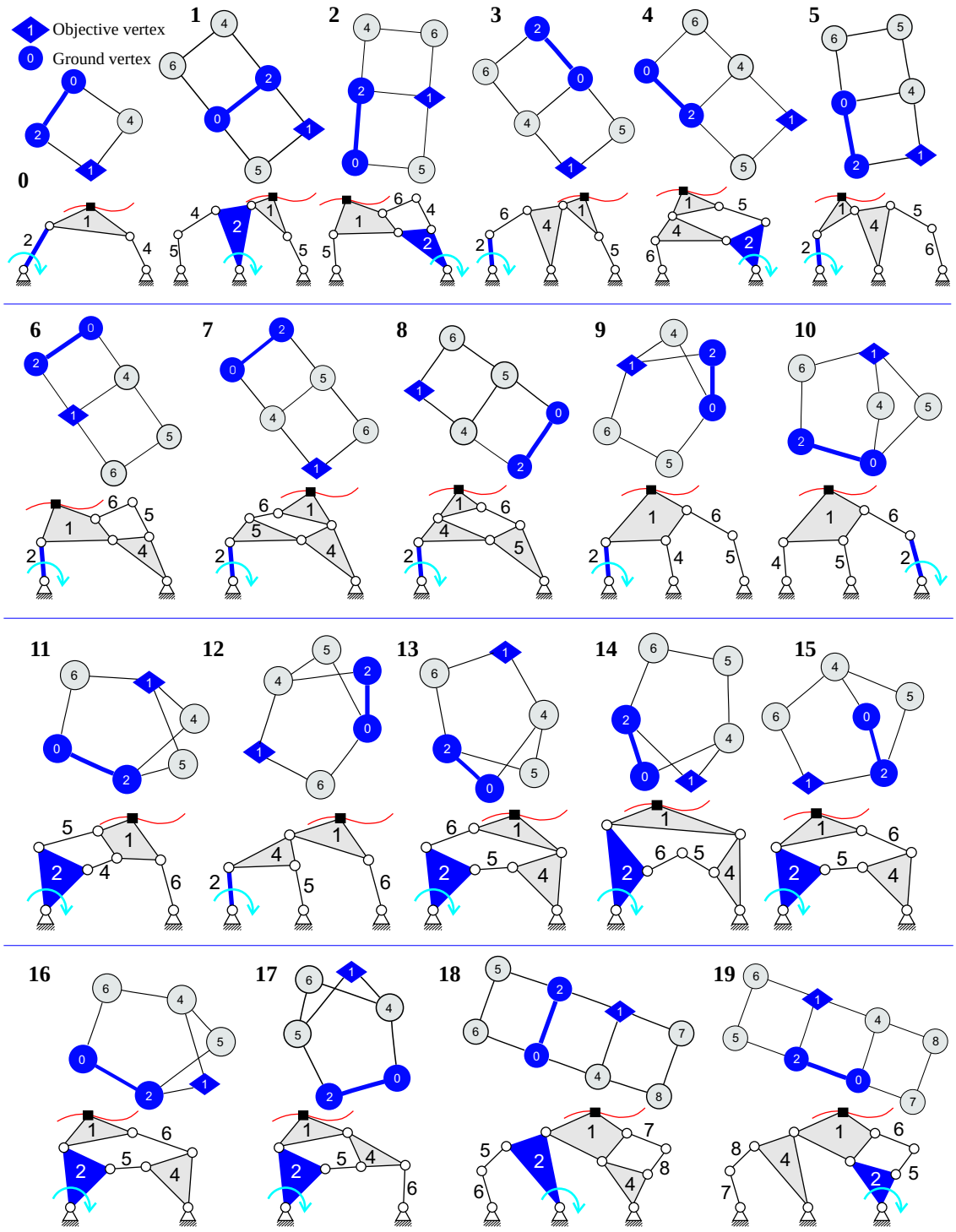


Figure 10: First 20 non-isomorphic occurrences of an initial graph for 4, 6 and the first two 8-bars options.

first 20 solutions are shown in Figure 10 and their respective codes are shown in Figure 11.

For instance, the four-bars solution has $A = 0$, $DC(S) = 50$, $\boldsymbol{d} = (2, 1)$, $\boldsymbol{C} = (1, 1, 1)$. We can also see that the method carries on the linkage inversion in an implicit way. In this example, solutions 2, 4, 6, 7 and 8 are Watt-I mechanisms; solutions 1, 3, and 5 are Watt-II mechanisms; solutions 9, 10 and 12 are Stephenson-I mechanisms; 11 , 16 and 17 are Stephenson-II mechanisms; and 13, 14 and 15 are Stephenson-III mechanisms.

Alternative	Mechanism	A	DC(S)	$\boldsymbol{d}$		$\boldsymbol{C}$		
				$d(v_1, v_0)$	$d(v_1, v_2)$	v_0	v_1	v_2
0	4-bars	0	50	2	1	1	1	1
1	6-bars	1	24851	2	1	1	1	2
2	6-bars	1	28818	2	1	2	1	1
3	6-bars	1	28818	2	3	1	2	2
4	6-bars	1	28818	3	2	2	1	2
5	6-bars	1	28818	2	1	1	2	2
6	6-bars	1	26976	2	1	2	2	1
7	6-bars	1	26976	2	3	2	2	2
8	6-bars	1	26976	3	2	2	2	2
9	6-bars	2	25876	2	1	1	2	1
10	6-bars	2	28818	2	2	1	3	1
11	6-bars	2	28818	2	2	3	1	1
12	6-bars	2	25876	2	2	1	2	3
13	6-bars	2	25876	2	2	2	1	3
14	6-bars	2	25876	2	1	2	1	2
15	6-bars	2	25876	2	1	2	1	3
16	6-bars	2	28818	2	1	3	1	2
17	6-bars	2	15168	2	2	3	3	2
18	8-bars	3	235282563	2	1	1	1	1
19	8-bars	3	235276426	2	1	1	1	1

Figure 11: First 20 solutions for a path generation with prescribed timing problem.

More examples were solved but results are not shown for space restrictions. The RBG problem of Figures 2 and 5 corresponds to the guiding of the slat of an airplane wing having 223 solutions with a prescribed distance from 2 to 3. The problem shown in Figure 6 is for synthesizing a landing gear retraction mechanism and has 231 alternatives in the atlas. Type and dimensional synthesis for both problems can be found in a previous work of the authors[24].

Remark 1 The number of permutations grows dominated by the factorial of the number of vertices in the initial graph. A search in the atlas of one four-bars KC, 2 six-bars KCs and 8 eigth-bars KCs, involves

$$\binom{4}{n_{ini}} n_{ini}! + 2\binom{6}{n_{ini}} n_{ini}! + 8\binom{8}{n_{ini}} n_{ini}! = \left[\binom{4}{n_{ini}} + 2\binom{6}{n_{ini}} + 8\binom{8}{n_{ini}}\right] n_{ini}!$$

permutations. For the Examples 1 and 2, where $n_{ini} = 3$, it is $[492]3! = 2952$. Next, for the Example 3 where $n_{ini} = 2$, there were $[260]2! = 520$ permutations explored. □

It should be mentioned that up to this point, we did not specify any particular type of joint. For example, either revolute or prismatic joints may be used in most joints for the graphs displayed in Figures 9 and 10. This aspect is solved next, in the specialization process for assigning joint types.

6 JOINT ASSIGNMENT: SPECIALIZATION

Three previous works gave foundations to the computer enumeration and assignment of link and joint types in the type synthesis of mechanisms:

- **Algorithm based on permutation groups:**

  Yan and Hwang[4] developed a method based on combinatorial theory for enumerating non-isomorphic specialized mechanisms precisely, starting from a specified kinematic chain. They developed an algorithm for finding the "permutation groups of a kinematic chain" (the link permutation group in conjunction with the joint permutation group) from its labeled link adjacency matrix. Based on these permutation groups they generated all non-isomorphic specialized mechanisms by assigning various types to the links and joints of the kinematic chain.

- **Algorithm based on the Characteristic Polynomial:**

  Murphy, Midha and Howell[25] developed a method for compliant mechanisms limited to binary pairs using the pseudo-rigid-body concept and defining the Compliant Segment and Compliant Connection vectors for enumeration. They also defined a Compliant Matrix (CM) where the diagonal entries represent the links/segments and outer diagonal entries represent the type of joints/edges. The polynomial expression used to find the eigenvalues of each CM, called characteristic polynomial $CP(CM) = |x\boldsymbol{I} - CM|$, is used to detect isomorphic mechanisms based on the theorem that states *"The adjacency matrices of two isomorphic kinematic chains possess the same characteristic polynomial (CP)"*[3].

- **Algorithm based in combinatorial analysis:** This approach was proposed by Freudenstein and Maki[28], and then extensively used by Tsai for joint assignment of kinematic chains, parallel platforms, geared automotive transmissions, robotic wrist mechanisms.

Our proposal is a combination of them, adding the "initial graph concept" to take account of the user defined constraints and also limit the combinatorial explosion. For instance, once G_{ini} is found as subgraph of G_a^p, we do not assign new types to joints that are connecting vertices inside of the isomorphic graph of G_{ini} in G_a^p, respecting the joint types defined by the user as prescribed parts. For this purpose, we first classify the joints in two groups: (i) those prescribed by the user, which are of fixed joints group, and (ii) those given by *specialization* using combinatorial analysis, which are added from a set of allowable types given by the user and positioned in the graph using some heuristic rules to be explained.

For adding prismatic joints, we can take into account singularities that can appear due to prismatic joint behavior and the number and orientation of prismatic joints over a link/circuit. Some

[3] Yan and Hall[26,27] have deeply studied the use of the CP for mechanisms. By counter examples Yan and also Tsai found that this theorem is a necessary but not sufficient condition for two KCs be isomorphic, so that the conversely is not true and two non-isomorphic KCs could share the same CP. Nevertheless, the CP is still in use for isomorphism testing purposes for KCs with less than 10 links.

topologies can be discarded while the enumeration takes place. This avoids future simulations over "a priory identifiable" unfeasible topology. Figure 12 shows a four-bars mechanism which all links have lost rotational DOFs because of the P-joints arrangement. In the case **a** one link (the coupler) has two P-joints with parallel directions, it has an undetermined position under a rotational motorization either on link 2 (see l_2) or 3. The remaining are immovable links. The case **b**, with three P-joints, has not mobility if a rotational grounded joint is motorized (see l_3 in **b-1**). Finally, in the case **c** only translational movements are obtained for any election of the motorized P-joint. Again the positions of links are undetermined.

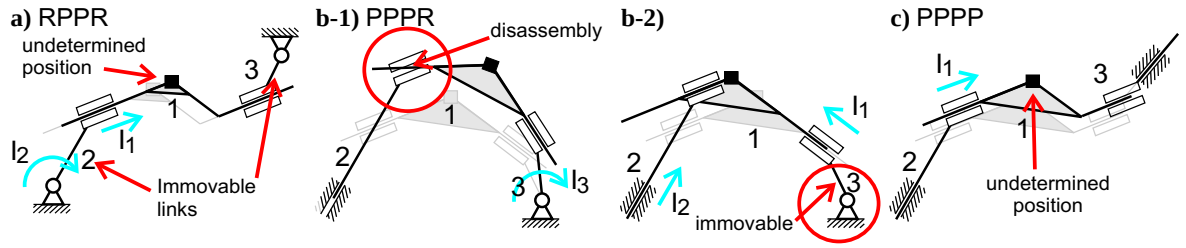


Figure 12: Uncompatible configurations over various four-bar mechanisms.

Remark 2 We have worked with one DOF mechanisms, so only one joint can be chosen as motorized. Note that if a P-joint pertaining to the coupler link is motorized, the result could be that obtained by a single controlled link (e.g. l_1 in case **a**, l_1 in case **b-2**). □

To avoid singularities, Sardain used a rule developed by Freudenstein and Maki[9] for P-joints assignment:

F&M: The maximum number of prismatic joints should be equal to one per link, except at the ground and the effector level.

Nieto[22] listed three restrictions:

N1: No link of a chain can contain only P pairs which directions are parallel.

N2: Binary links of a chain with only P pairs cannot be connected directly.

N3: No closed circuit of a chain can have less than two R pairs.

These rules are heuristic, and not fully compatible. In fact, rule **F&M** is more restrictive than **N1**, **N2**, and **N3**. Since the **N1** rule can be computed only after the dimensional synthesis stage, in our algorithm we use the rules **N2**, and **N3** for P-joints assignments.

In this way, we state a specialization problem with restrictions due to user prescriptions and to the particular nature of joints. The algorithm starts considering that all links are constrained by R-joints. Then, it tries to add a given number of P-joints n_P to explore if new and better alternatives are generated.

6.1 Synthesized joints: added joints group

For a given topology, when we make the specialization of the added joints group, there appears again the problem of isomorphism detection but now due to joint types. A systematic strategy for identification, enumeration and retrieval of alternatives is needed.

We define a colored adjacency matrix with the complete information of the topology. Following Murphy[25], we put link characteristics on the diagonal entries and the joint characteristics on the outer diagonal ones, but consider the entries in a different way.

The colored adjacency matrix is filled as follows: $A_{ii} = 1$ if the vertex i pertain to the initial graph (including the ground), $A_{ii} = 2$ if the vertex i is the objective vertex and, $A_{ii} = 0$ if i is a type synthesized vertex. The joint types are represented by the symmetric entries, $A_{ij} = 1$ for rotoidal joint type, $A_{ij} = 2$ for prismatic joint type, and $A_{ij} = 0$ if no connection appears. Summarizing, we define:

$$A_{ii} = \begin{cases} 1 & \text{if} \quad v_i \in G_{ini}, \\ 2 & \text{if} \quad v_i = v_{obj}, \\ 0 & \text{otherwise}, \end{cases} \qquad A_{ij} = \begin{cases} 1 & \text{Rotoidal joint}, \\ 2 & \text{Prismatic joint}, \\ 0 & \text{no connection}. \end{cases}$$

Let E_a be the set of the added joints group $E_a = E(G_A^p) - E(G_{ini})$, and be n_a the number of synthesized joints ($n_a = |E(G_A^p)| - |E(G_{ini})|$).

To limit the search, the user can define a prescribed number of P-joints n_P to add in E_a. The number of combinations for adding exactly n_P P-joints is $\binom{n_a}{n_P}$. For classification, all n_P-combinations of P-joints in the set E_a are assigned. For a given combination i, the degree code in base 3 $DC_3^d(A_i)$ is computed. The solution is retained only if the degree code is different from all those already stored.

Once the enumeration is finished, the code of the colored topology DC_3^d can also be used for retrieving a solution. See Appendix B.

Example 3 Joint assignment for a RBG task

Consider the problem of adding P-joints to a mechanism suited to develop the rigid body guidance task shown in Figure 5. From the Number Synthesis stage, the simplest feasible topology is a four-bars mechanism. In Figure 13 the solutions for adding from one to four P-joints are shown. This is not completely useful, because it could have kinematic singularities due to the joint nature. Applying the Nieto rules the last four alternatives are eliminated.

In this problem the fixed group is empty $|E(G_{ini})| = \emptyset$, because there are not prescribed joints.

By adding one P-joint, we get the two first configurations shown in Figure 13 with codes stored in base three 48172, and 54706. In spite of their topological similarity, their physical meaning is different. See the sketches below each graph.

For the assignment of two P-joints we have $\binom{4}{2} = 6$ possible combinations. There appears four non-isomorphic solutions, their respective codes are: 48175, 54733, 54709, and 56893.

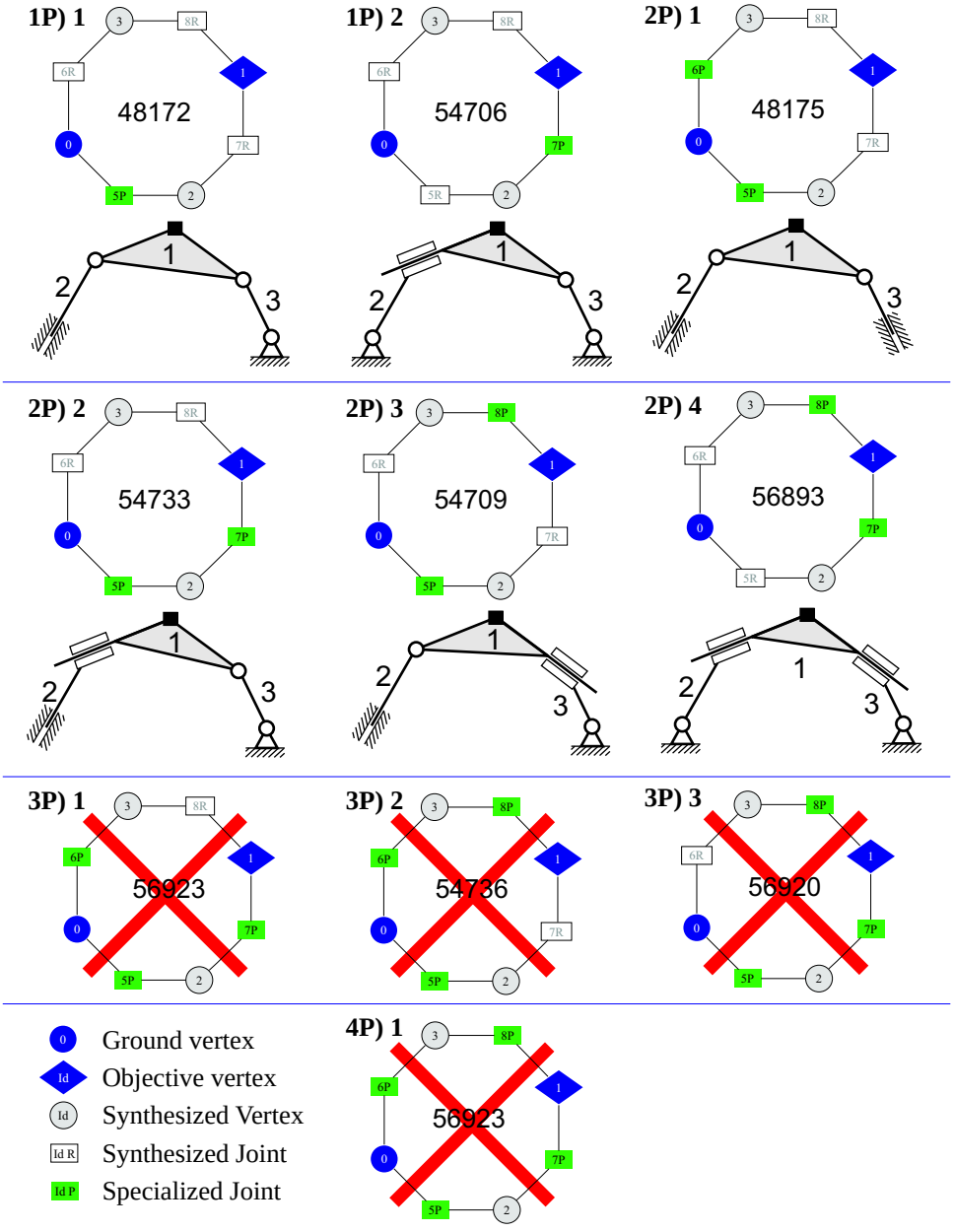


Figure 13: Example of non isomorphic P-joints assignments on synthesized joints.

7 CONCLUSIONS

Tsai based all his systematic enumeration methodologies on the Feudenstein and Maki concept of "separation" of kinematic structure (to generate alternatives) from function (to evaluate the generated alternatives). He also used to remark *"The more functional requirements that are translated into structural characteristics and incorporated in the generator, the less work is*

needed from the evaluator. However, this may make the generator too complex to develop. Generally, if a functional requirement can be written in a mathematical form, it should be included in the generator".

In this work we have experienced his words since we have "integrated" more structural characteristics using the initial graph concept, developing a more complex generator. This diminishes considerably the time computation in all subsequent stages of design. For the next dimensional synthesis stage the information of the known nodes and the type-synthesized graph structure could be used to apply either a general absolute/natural coordinate formulation or the complex number formulation based on the decomposition of the topology into open chains (Sandor[1], Cardona[18], Pucheta and Cardona[24]).

The main contribution of this work is to offer a systematic procedure to obtain topological alternatives for a given kinematic problem. New algorithms based on Graph theory and combinatorial analysis were developed to search and codify the solutions in a non isomorphic way.

Other remarkable characteristics are:

- The use of an enumerated atlas assures that all candidate mechanisms satisfy the required degree of freedom and does not contain rigid subchains.
- The number of solutions is finite and the combinatorial explosion is manageable.
- The designed Diagonally Extended Degree Code allows coding and decoding of solutions in an efficient and straightforward way.

Murphy *et al.*[25] and also Howell[25,29] prove that rigid-body enumeration is an essential starting for type synthesis of compliant mechanisms.

The algorithms are useful for dealing with more types of links and joints in the mechanism and even more complex tasks. However, like we seen for prismatic joints, adequate rules to reject those kinematically invalid solutions must be properly designed.

8 ACKNOWLEDGMENTS

This work has received financial support from *Consejo Nacional de Investigaciones Científicas y Técnicas*, *Agencia Nacional de Promoción Científica y Tecnológica*, *Universidad Nacional del Litoral* (CONICET, ANPCyT and UNL, from Argentina) and from the *European Community* through grant *SYNCOMECS* (SYNthesis of COmpliant MEChanical Systems) project UE FP6-2003-AERO-1-516183.

REFERENCES

[1] G.N. Sandor and A.G. Erdman. *Advanced Mechanism Design: Analysis and Synthesis*, volume 2. Prentice-Hall, New Jersey, (1984).

[2] A.G. Erdman. Computer-aided design of mechanisms: 1984 and beyond. *Mechanism and Machine Theory*, **20**, 245–249 (1985).

[3] D.G. Olson, A.G. Erdman, and D.R. Riley. A systematic procedure for type synthesis of mechanisms with literature review. *Mechanism and Machine Theory*, **20**, 285–295 (1985).
[4] H.S. Yan and Y.W. Hwang. The specialization of mechanisms. *Mechanism and Machine Theory*, **26**, 541–551 (1991).
[5] Lung-Wen Tsai. *Mechanism Design: Enumeration of Kinematic Structures According to Function*. CRC Press, Boca Raton, (2001).
[6] A. S. Hall Jr. *Kinematics and Linkage Design*. Waveland Press, U.S.A., (1961).
[7] C.S. Lin, A.G. Erdman, and B.P. Jia. Use of compatibility linkages and solution structures in the dimensional synthesis of mechanism components. *Mechanism and Machine Theory*, **31**, 619–635 (1996).
[8] J.M. McCarthy. *Geometric Design of Linkages*. Springer, (2000).
[9] P. Sardain. Linkage synthesis: Topology selection fixed by dimensional constraints, study of an example. *Mechanism and Machine Theory*, **32**, 91–102 (1997).
[10] A.G. Erdman and G.N. Sandor. *Mechanism Design: Analysis and Synthesis*, volume 1. Prentice-Hall, New Jersey, 3rd edition, (1997).
[11] M.R. Hansen. A multi level approach to synthesis of planar mechanisms. In *Computer Aided Analysis of Rigid and Flexible Mechanical Systems*, (1993).
[12] F. Harary. *Graph Theory*. Addison-Wesley Series in mathematics, (1969).
[13] D.B. West. *Introduction to Graph Theory*. Prentice Hall, (2001).
[14] A. Cardona, I. Klapka, and M. Géradin. Design of a new finite element programming environment. *Engineering Computations*, **11** (1994).
[15] Open Engineering S.A. *OOFELIE: Oriented Object Finite Elements Led by Interactive Executor*. http://www.open-engineering.com.
[16] W-M Hwang and Y-W Hwang. Computer-aided structural synthesis of planar kinematic chains with simple joints. *Mechanism and Machine Theory*, **27**, 189–199 (1992).
[17] H-I Hsieh. Sistematic methodologies for the automatic enumeration of topological structures of mechanisms. Master's thesis, University of Maryland, USA, (1992).
[18] A. Cardona. *Computational Methods for Synthesis of Mechanisms*. Technical report, CIMEC-INTEC, (2002).
[19] Y-M Moon and S. Kota. Automated synthesis of mechanisms using dual vector algebra. *Mechanism and Machine Theory*, **37**, 143–166 (2002).
[20] S.J. Chiou and S. Kota. Automated conceptual design of mechanisms. *Mechanism and Machine Theory*, **34**, 467–495 (2004).
[21] Y-X Wang and H-S Yan. Computerized rules-based regeneration method for conceptual design of mechanisms. *Mechanism and Machine Theory*, **37**, 833–849 (2002).
[22] J. Nieto Nieto. *Síntesis de Mecanismos*. Editorial AC, Madrid, (1977).
[23] C.S. Tang and T. Liu. The degree code - a new mechanism identifier. *ASME Journal of Mechanical Design*, **115**, 627–630 (1993).
[24] M.A. Pucheta and A. Cardona. Type synthesis and initial sizing of planar linkages using graph theory and classic genetic algorithms starting from parts prescribed by user. In *Multibody Dynamics 2005, ECCOMAS Thematic Conference*, (2005).

[25] M.D. Murphy, A. Midha, and L.L. Howell. The topological synthesis of compliant mechanisms. *Mechanism and Machine Theory*, **31**, 185–199 (1996).
[26] H.S. Yan and A.S. Hall. Linkage characteristic polynomials: Definition, coefficients by inspection. *ASME Journal of Mechanical Design*, **103**, 578–584 (1981).
[27] H.S. Yan and A.S. Hall. Linkage characteristic polynomials: Assembly theorem, uniqueness. *ASME Journal of Mechanical Design*, **104**, 11–20 (1982).
[28] F. Freudenstein and E.R. Maki. Creation of mechanisms according to kinematic structure and function. *Journal of Environmental and Planning B*, **6**, 375–391 (1979).
[29] L.L. Howell. *Compliant Mechanisms*. John Wiley & Sons, New York, (2001).

A PERMUTATIONS

A *permutation* of n distinct elements $x_1, \ldots, x_n$ is an ordering of the n elements $x_1, \ldots, x_n$. By the Multiplication Principle, there are

$$n(n-1)(n-2)\cdots 2 \cdot 1 = n!$$

permutations of n elements.

A *r-permutation* of n (distinct) elements $x_1, \ldots, x_n$ is an ordering of an r-element subset of $\{x_1, \ldots, x_n\}$. The number of r-permutations of a set of n distinct elements is denoted $P(n, r)$, where

$$P(n,r) = n(n-1)(n-2)\cdots(n-r-1) = \frac{n!}{(n-r)!}, \qquad r \leq n.$$

A selection of n distinct elements $x_1, \ldots, x_n$ without regard to order is called a *combination*. A *r-combination* of a given set $X = \{x_1, \ldots, x_n\}$ is an unordered selection of r-elements of X. The number of r-combinations of a set of n distinct elements is denoted $C(n, r)$ or $\binom{n}{r}$, where

$$C(n,r) = \frac{P(n,r)}{r!} = \frac{n!}{(n-r)!r!}, \qquad r \leq n.$$

In permutations and combinations used in this work repeated elements are not allowed. For instance, we generate all r-permutations of an n-element set to explore subgraphs inside a graph. Sets with unordered and repeated elements concern to Generalized Permutations and Combinations Theory and will be used in future for more than one type of links and joints.

B RETRIEVING MECHANISMS FROM INTEGER CODES

Given a DC_b^d, and the base b of a KC we can retrieving the a stored mechanism. For instance, for R and P joints we need a base three, so $b = 3$.

Let DC_3^d a degree code stored in base 3, for the $n \times n$ adjacency matrix A of a graph $G(E, V)$ with n vertices and e edges.

The matrix A has a number of m significative entries (including the diagonal). To find it we

made

$$3^m = DC_3^d; \quad \log_{10}(3^m) = \log_{10}(DC_3^d); \quad m = \left\lceil \frac{\log_{10}(DC_3^d)}{\log_{10}(3)} \right\rceil$$

In general,

$$m = \left\lceil \frac{\log_{10}(DC_b^d)}{\log_{10}(b)} \right\rceil$$

To find the size n of A, we know that

$$\frac{n \times n - n}{2} + n = m; \quad \frac{n^2 + n}{2} = m$$

this leads to solving the quadratic expression

$$n^2 + n - 2m = 0$$

so, n is

$$n_{1,2} = -\frac{1}{2} \pm \frac{1}{2}\sqrt{8m+1}$$

and rounded the higher solution to the nearest integer we have

$$n = \left\lceil -\frac{1}{2} + \frac{\sqrt{8m+1}}{2} \right\rceil.$$

Knowing DC_b^d (`code`) and b (`nb_colors`) we can also identify the number of edges en the types of each edge using an algorithm for computing the digits in b-base of the DC_b in 10-base. To give an idea, the algorithm in C++ code is

```
int d,u,i,j,coef,e=0;
float base = (float)nb_colors+1;
int m = ceil(log10((float)code)/log10(base));
int n = ceil((-1.0+sqrt(8.0*m+1))/2.0);
Graph A(n); // adjacency matrix
d = n;      // diagonal index
u = m-n;    // upper diagonal index
std::vector <int> links (d);
std::vector <int> joints (u);
e=0;
for (i=n-1;i>=0;i--)
    for (j=n-1;j>=i;j--) {
        coef =fmod(code,base);
        if (i==j) links[--d] = coef;
        else{
            joints[--u] = coef;
```

```
            if (coef){
                A.Connect(i,j);
                e++;
            }
        }
        code -= coef;
        code /= base;
    };
```

where `Graph` is a class of graphs and the member function `Connect` sets simultaneously a `1` in the `(i,j)` and `(j,i)` entries. The number of vertices/links is `n`, the number of edges/joints is `e`. We get the adjacency matrix in `A`, the links types in `links`, and the joint types in the `e` non-null entries of `joints`. Thus, the mechanism is completely characterized. To save memory, the adjacency matrix is stored as a set of $m - n$ bits, the link type as a vector of `n` integers, and then the joint types as a vector of `e` integers.

C ALTERNATIVE DEGREE CODE

An integer is represented by 32 bits and the maximum "unsigned integer" is $2^{32} = 4294967295$. The *limit* for two colors is:

$$2^{32} = 3^m$$
$$m = \left\lfloor \frac{32 \log_{10}(2)}{\log_{10}(3)} \right\rfloor = 20.$$

With the previously presented DC_3^d we violate this limit at using six-bars topologies

$$n = 6 \rightarrow m = 21$$

.

To save this inconvenient, the degree code could also be modified to be stored as a list of integers representing each row of the upper sub matrix of the adjacency matrix including the diagonal elements. We denote it Diagonally Extended Degree Code DC_b^{dr} by Rows (in base b).

$$A_1 = \begin{bmatrix} 2 & 0 & 1 & 1 \\ & 1 & 1 & 2 \\ & & 0 & 0 \\ & & & 0 \end{bmatrix}, DC_3^{dr}(A_1) = DC_3^{dr}\left(\begin{bmatrix} 2 & 1 & 1 & 0 \\ & 0 & 0 & 2 \\ & & 0 & 1 \\ & & & 1 \end{bmatrix}\right) = \begin{bmatrix} (2110)_3 \\ (002)_3 \\ (01)_3 \\ (1)_3 \end{bmatrix} = \begin{bmatrix} 66 \\ 2 \\ 1 \\ 1 \end{bmatrix}_{10}$$

The capacity for represent colored adjacency matrices is improved. We are covering three colored matrices up to 20×20 and therefore mechanisms of 20 links (or matrices of 16×16 with codes in base 4 thinking in a nearly future extension to flexible mechanisms). On the other hand, the number of comparisons needed for isomorphism testing is augmented, and also, the required memory to store a code grows $n-1$ times. The number of comparisons can be reduced

exiting at the first different occurrence between two integers in the same position. For instance, if we have another colored adjacency matrix

$$A_2 = \begin{bmatrix} 0 & 2 & 0 & 1 \\ & 2 & 1 & 0 \\ & & 0 & 1 \\ & & & 1 \end{bmatrix}, DC_3^{dr}(A_2) = DC_3^{dr}\left(\begin{bmatrix} 2 & 2 & 1 & 0 \\ & 0 & 0 & 1 \\ & & 0 & 1 \\ & & & 1 \end{bmatrix}\right) = \begin{bmatrix} (2210)_3 \\ (001)_3 \\ (01)_3 \\ (1)_3 \end{bmatrix} = \begin{bmatrix} 75 \\ 1 \\ 1 \\ 1 \end{bmatrix}_{10}$$

beginning by the first row of their codes, we can easily see, for this case, that we need only one comparison ($66 < 75$) for isomorphism testing

$$DC^{rd}(A_1) = \begin{bmatrix} 66 \\ 2 \\ 1 \\ 1 \end{bmatrix} \quad < \quad DC^{rd}(A_2) = \begin{bmatrix} 75 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

so the two matrices represent two non-isomorphic mechanisms.

NOMENCLATURE

BKC: Basic Kinematic Chain

KC: Kinematic Chain

F: degree of freedom of a mechanism

n: number of links in a mechanism

j: number of joints in a mechanism, assuming that all the joints are binary

P: prismatic joint, also called *sliding pair*

R: revolute joint, also called *turning pair*, a *hinge* or a *pin joint*

f_i: degrees of freedom associated with joint i

L: number of independent loops in a mechanism

λ: freedom of the space in which a mechanism is intended to function

$\boldsymbol{N}$: set of nodes taken from CAD interface

$\boldsymbol{F}$: set of fixations on nodes taken from CAD interface

$\boldsymbol{E}$: set of elements (rigid bodies, hinges, prismatic joints, etc.) taken from CAD description

$\mathcal{D}$: set of displacements prescribed on nodes of parts

$\mathcal{L}$: set of orientations prescribed on nodes or elements of parts

$\mathcal{J}$: set of translational or rotoidal motorizations prescribed on P or R joints

$\mathcal{M}$: mechanism class composed by $\{\boldsymbol{N}, \boldsymbol{F}, \boldsymbol{E}\}$, G, $\mathcal{D}$, $\mathcal{L}$, and $\mathcal{J}$.

$G(E, V)$: graph of a kinematic chain composed by a set of edges E, a set of vertices V, and a adjacency matrix A describing their interconnections.

A: adjacency matrix

e_i: edge of a graph G

v_i: vertex of a graph G

$d(v_i)$: degree of the vertex i, or number of vertices connected to the vertex i

$d(v_i, v_j)$: distance between vertex i and vertex j

$C(v_i)$: vertex class

p: permutation vector

$DC(A_i)$: Degree Code of the adjacency matrix i

$DC_b^d(A_i)$: Diagonally Extended Degree Code of the adjacency matrix i in base b.

$DC_b^{dr}(A_i)$: Diagonally Extended Degree Code of the adjacency matrix i in base b computed by rows.