Proceedings of the ASME 2015 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2015
August 2-5, 2015, Boston, Massachusetts, USA

DETC2015-46073

AUTOMATED SYNTHESIS OF PLANAR MECHANISMS WITH REVOLUTE, PRISMATIC AND PIN-IN-SLOT JOINTS

Weifeng Huang
Mechanical Engineering
Oregon State University
Corvallis, Oregon
huangwe@onid.oregonstate.edu

Matthew I. Campbell
Mechanical Engineering
Oregon State University
Corvallis, Oregon
matt.campbell@oregonstate.edu

ABSTRACT

This paper presents a graph synthesis approach to planar N-bar mechanisms with revolute (R), prismatic (P), and RP (pin-in-slot) joints. This novel graph synthesis method extends the enumeration to define the possible topologies for mechanisms with any mix of R-, P-, and RP-joints. Each topology is explicitly defined as a graph which can be viewed and simulated within an online kinematic simulator. This method successfully produces the same number of topologies for 6, 8, 10, and 12-bar revolute joint mechanisms as those shown in the existing literature. It explicitly calculates all topologies for 6, 8, and 10 bars with a mix of the three joint-types.

INTRODUCTION

It is well known that planar mechanisms like four bars and cranks-and-sliders have been used for centuries to accomplish different tasks. While many planar mechanisms are common knowledge to mechanical engineers, the mechanisms can be highly complex and innovative. Planar mechanisms – whether they are simple or complicated – are all part of the same language follows the same restrictions. Since Gruebler's equation governing the mobility of planar mechanisms is straightforward to solve (with simply the number of links and joints), it would appear possible that the entire valid space of 1 degree-of-freedom mechanisms can be captured explicitly. In this space, each of the linkages is controlled by a single input. Meanwhile, linkages that contain rigid structures or static links are considered as violations and removed from this space. Various other authors [1][2][3][4] have attempted such endeavors and we are building upon their accomplishments and our previous accomplishments to represent all planar mechanism topologies as a library of graphs. As the number of links and joints in a linkage increase, the complexity of the paths also increases – beyond what is produced by common 4 and 6 bar linkages. This is important to help engineers explore new possibilities for accomplishing complex kinematic tasks.

In recent decades, the focus on robotic systems has led many to pursue multi-actuator systems for complex movements that could have been accomplished by much simpler one degree-of-freedom mechanisms. Although it is not substantiated, such single degree of freedom mechanisms may prove to be lighter, more robust, and more efficient than high degree-of-freedom robotic "open chains." Regardless, exploring the space of planar mechanisms has always been challenging. With a repository of mechanisms, engineering designers could more easily explore the space of possibilities. Furthermore, it is possible that computational search could optimize the dimensions of one or more of the many valid topologies to best meet the user's desired kinematic behavior.

In this paper, a formal and implemented approach is developed to capture the valid space of planar mechanisms. This approach not only synthesizes revolute joint linkages as shown by previous authors; it additionally includes prismatic (P) and pin-in-slot (RP) joints. This is accomplished by a set of graph transformations – referred to as grammar rules – that construct graphs in the systematic method presented by Tsai [5]. The rules build upon simpler graphs by adding links and joints in a structured way. The rules are constructed to avoid creating degenerate mechanisms. The rules also transform simpler dyadic mechanisms into more complex linkages, like the double butterfly mechanism, that lack any dyadic chains.

One challenge of the study is to synthesis linkages includes prismatic (P) and pin-in-slot (RP) joints. As an example to illustrate the challenges with P- and RP-joints, consider the modified four-bar linkage in Fig. 1, which has one input revolute and three prismatic joints. Such a mechanism will have no rotation about the R-joint due to the lack of complementary

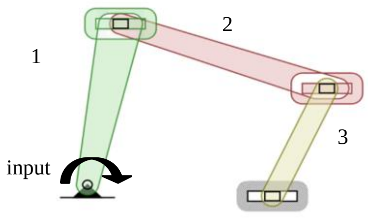


FIGURE 1: A 4 bar linkage with 3 parallel prismatic joints.

rotation in the remaining P-joints. As shown with three parallel slot P-joints, the movement of links 2 and 3 are independent. If distinct angles were given to the P-joints, there would still be 1 degree-of-freedom in the sliding members and no rotation of the R-joint. While such a degenerate mechanism and its generalization are known, it is described here to give example to the types of constraints that need to be added to the graph synthesis approach to avoid infeasible mechanisms.

In this study, a synthesis method first generates all valid 1-DOF R-joint linkages. One or more of these R's are replaceable by P- and RP-joints following certain rule restrictions. Since Gruebler's equation may lead to some violations, geometry and graph connectivity need to be considered when P- and RP-joints are involved in the enumeration of topologies. As in the aforementioned example, the DOF of the modified 4-bar linkage with 3 P-joints is equal to 1 when applying Gruebler's equation. In Gruebler's equation:

$$3(L-1) - 2J_1 - J_2 = M \qquad (1)$$

L represents the number of links or rigid bodies, J_1 represents the number of one degree-of-freedom joints (such as R and P), and J_2 represents the two degree-of-freedom joints (such as RP and G). The result M, is also known as the overall mechanism degrees-of-freedom. So the DOF for the linkage in Fig. 1 is calculated as below:

$$3*(4-1) - 2*4 - 0=1 \qquad (2)$$

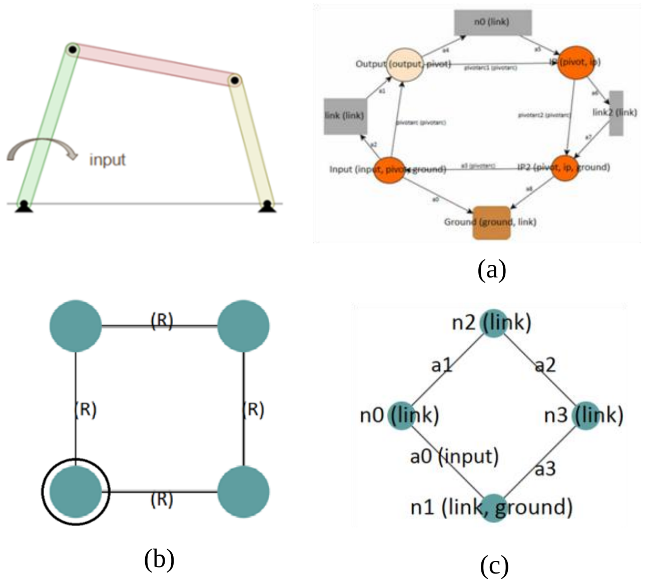
But in reality, the mobility is 2 due to the parallel slot P-joints in link 2 and 3.

In the remainder of this paper, we present other approaches in the literature that seek to define the space of valid planar mechanisms Sections 2 and 3 followed by a detailed discussion of our graph transformation approach Section 4. The results that are generated by using this approach are shown in Section 5. The conclusions of the paper Section 6 discuss the results and additional work.

RELATED WORK

There are many researchers who have been working on exploring the design space of planer mechanisms, and the results for 6, 8 and 10 bar mechanisms with only revolute joints are well established [1][2][5][6].These publications present the permutations of the possible linkages, but contradict when the linkages are greater than 12 bars[2][3].

When the linkages synthesis involves P-joints, Sardain provides a method to avoid the mobility being affected by P-joints [7]. In his research, he divides a linkage into two different kinds of components: primary components and secondary components. The primary components are those components that can be calculated from the dimensions at the beginning, and the synthesis of secondary components, which depend on the results from primary components. If a link with a P-joint needs to be added to the primary components, the link must be a binary link. Also, Sardain set the limit that it can only have one P-joint in each link. Based on these rules, 43 topologies for 4 and 6 bars that contain P-joints are found. Here, we show that the space with P-joint is significantly bigger.



(a)

(b)

(c)

FIGURE 2: Three different graph representations for a single input 4 bar linkage.

In order to achieve this exploration, the transformation from graph to real linkage is based on Tsai's graph representation of linkages. In his research, complex mechanisms are represented by very simple graphs. This is very valuable since this study is based on graph-based search, and it provides a connection between graphs and real mechanisms.

FOUNDATIONS OF GRAPH REPRESENTATION

In order to automate the synthesis of planar mechanisms, we need to find an appropriate way to represent them by graphs. Following some representation scheme, rules for constructing these graphs will need to be defined. Two different graph representations are discussed in this paper.

In Radhakrishnan's study [8], the graph representation includes many details. For a four bar linkage in Fig. 2a, local labels indicate the function of nodes and arcs. For example, the local label *link* represents a link or rigid body and *pivot* represents an R-joint which connects two links. While concrete objects are represented by nodes, the arcs are used only to represent the flow of energy.

The other graph representation investigated is by Tsai [5]. Here in Fig. 2b, nodes indicate links in the linkage, and arcs indicate the joints. The prime mover is indicated by a node that has an additional circle around it. Local labels only appear in arcs to show the type of joints. Since this graph representation is used for structural analysis, physical dimensions of the linkages are not considered.

Radhakrishnan's approach is more descriptive than Tsai's because it includes additional elements needed to define a mechanisms. By representing the joints as nodes, two aspects of the mechanism can be more clearly defined. First, since nodes are typically attributed to Cartesian coordinates in representing graphs, the joints can be explicitly positioned by the corresponding joint node. Second, the limitation of the arc-as-joint in Tsai's representation prevents representing three or more links joined by the same joint. Radhakrishnan's can explicitly represent this as its own topology.

In this study, the graph representation is based on Tsai's work, with some modifications towards Radhakrishnan's approach. In Fig. 2c, local labels are used to describe joints as is done in Tsai's but are also used to indicate the *ground* link. The default joint type is revolute, so instead of including an "R" label, it is left blank. Additional labels for "P" and "RP" represent prismatic sliding joints and pin-in-slots half-joints respectively. Tsai never explicitly captures the RP-joint but rather sees this as a special case when an R and a P occupy the same location. Our approach explicitly captures half-joints as unique topologies since the goal is to synthesize realizable mechanisms from the graph representation. Tsai developed his approach more as a method of analysis and categorization, so the avoidance of RP is justified. Interestingly, Tsai does introduce the gear, or "G" half joint, so his method does not completely avoid the complications caused by these interactions. When a linkage contains RP-joints, there is an added complexity due to a lack of symmetry in the joint. One of the links serves as the slide or P component of the RP-joint, while the other link serves as the R, revolute, or pin in the pin-in-slot. This is worth distinguishing give the significant effect on the kinematics of the linkage. For example in Fig. 3, two linkages have the same topologies and dimensions, but the direction of the RP-joint are different. The path indicates the movement of a tracer point on the ternary plate; and it is clear that the paths are different on the left and right mechanisms. On the left of the figure, the RP-joint is represented as an arc going from ground to the ternary link. With the tail of the arc corresponding to R and the head as the P, the R is on ground and the slide is part of the ternary plate. On the right side mechanism, the arc direction is switched. In our augmentation of Tsai's representation, arc direction will be used to indicate different mechanisms topologies. These figures and many others in the paper are generated from the Planar Mechanism Kinematic Simulator which is available by the author [9]. Note that the references in this paper include many persistent URLs to complete simulations and animations of the generated mechanisms.

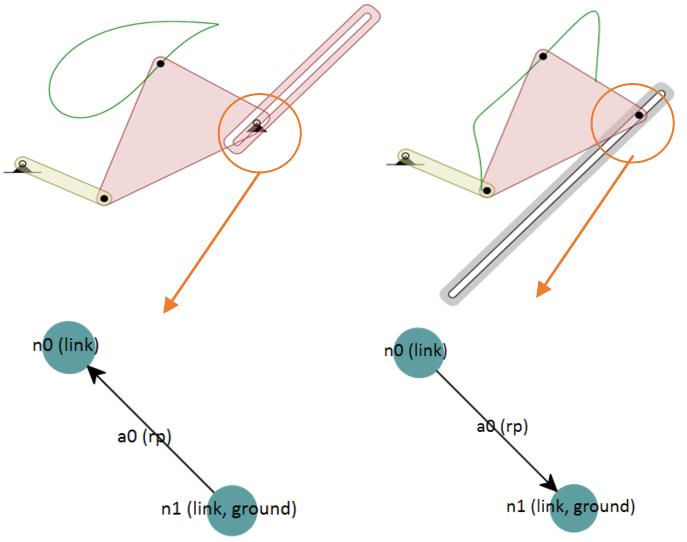


FIGURE 3: Two RP-joint arcs with different direction of arrow represent the order of the pin and slot in the real linkages [10][11].

GRAPH SYNTHESIS PROCESS

With an established graph representation, we can now start to explore topologies by using graph transformation approach. This graph transformation process starts from a host graph. After applying graph grammar rules to the host, a new graph is created. This methodology is described in Campbell's study [12], in which graph grammar rules serve as a production rule system to create a tree of graph topologies.

The function of graph grammar rules is to modify the graph by adding or removing nodes, arcs and/or labels. Then a new graph is generated. Accordingly, the function of graph grammar rules is to add components or make modifications to an existing linkage resulting in a new valid linkage. Figure 4 shows an example of applying a dyad rule to a pendulum and creating a 4-bar linkage. The two nodes in the host graph *n0* and *n1* are matched to the two nodes *n0* and *n6* in the left hand side of the rule, which means *n0-n1* is a valid location for applying this dyad rule. This matching process is referred to as graph recognition. After that, the host graph is modified according to how the right hand side of the rule differs from the left. In this example, the right hand side of the rule retains nodes *n0* and *n6* from left hand side with a new addition (a0, n1, a1, n2 and a2). This addition is referred to as a dyad component which has 2 links and 3 joints, and it is added to location *n0-n1* which was identified as a valid mapping of *n0-n6* from the rule. In this study, different grammar rules are used to encapsulate the constraints of planar mechanisms and define how more complex mechanism graphs can be created from simpler ones.

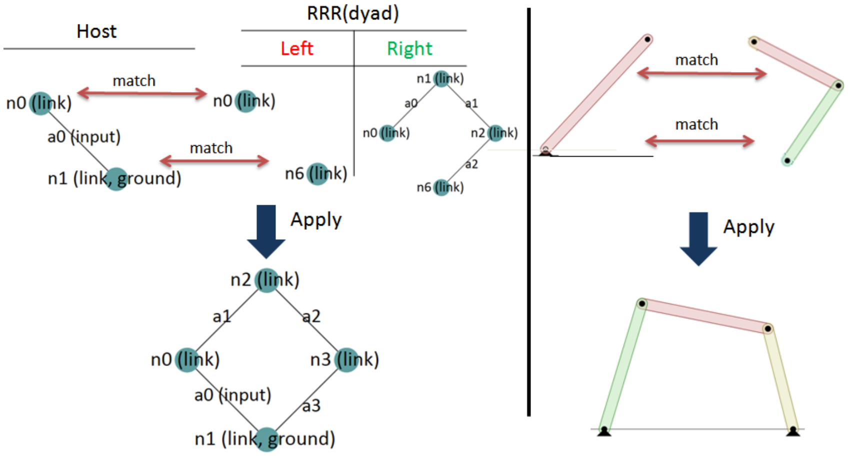


FIGURE 4: An example of applying a dyad rule to generate a 4 bar linkage with its representation in real linkage design.

Overview

This collection of grammar rules can explore the full design space and identify and remove rigid structures from the results. The rules for this searching process are divided into four categories based on their functions in synthesis process: generation, transformation, detection and joint change.

The function of generation rules is to add components to increase the number of links and joints in the linkage. After that, transformation rules are applied to relocate the ground joint. The detection rules can recognize and extract rigid structures. Finally, the joint change rule will recognize R-joints and replace them by P-joints when the position is accessible. The details of these rules are discussed next.

Generation rules

The function of generation rules is to add components to the linkage without changing its mobility. The construction of these components is based on Gruebler's equation with new variables as below:

$$3(L\text{-}1+L_{rule}) – 2(J_1 + J_{1ruel}) – (J_2 + J_{2rule}) = M \qquad (3)$$

In this equation, variables L_{rule}, J_{1ruel} and J_{2rule} indicate the number of links, 1-DOF joints and 2_DOF joints that could be added to the linkage. The number combinations of these 3 variables can be found to ensure that the mobility M is conserved. Since RP-joints are addressed later, J_2 *and* J_{2rule} are set to 0 We focus on possible L_{rule} and J_{1ruel} combinations to provide insight into the valid permutations of generation rules. Also, each of the permutations is depicted in Table 1 by a mechanism substructure. Some examples are shown in Tab. 1. Given that the possible additions increase with multiples of the dyad, we do not need to consider these higher additions unless they define topologies that would not be achieved by repeated calls of the dyad rule (as described in Fig. 4). For the dyadic rule, it can recognize any pair of different links as part of the left-hand-side. For the more complex non-dyadic rules, only one of the recognized links is in the mechanism with the others attached to the ground. This simplifies the generation constraints but limits the possible configurations that are created. To solve this, the next transformation rule explores the higher linkage mechanisms to fully cover the space of possibilities.

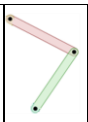
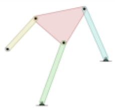
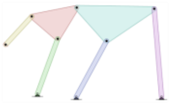
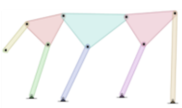
Transformation rule: Move Joint from Ground to Link

A transformation rule is needed because the generation rules do not create all the possible configurations. The transformation rule detaches a joint from one link and reattaches it to a different link to create new mechanism. The transformation rule is only valid when the number of ground joints is higher than two. One of the non-input ground joints is disconnected from ground and connected to another link to effectively convert dyads to more complex coordinated links. A single application of this rule generates a double butterfly linkage as shown in Fig. 5. First, the linkage in Fig 5.a is generated by adding double-triad to the single pendulum L_1. After that, the transformation rule can be applied to detach the ground joint J_1 and connect it to L_1 resulting in the double butterfly linkage in Fig. 5b.

Taboo Detection Rules

Although the transformation rule can add new configurations to the design space by modifying the graph, it

TABLE 1: Components could be added to the linkage without changing its mobility.

				
	Dyad	Triad	Double-Triad	Triple-Triad
L_{rule}	2	4	6	8
J_{1rule}	3	6	9	12

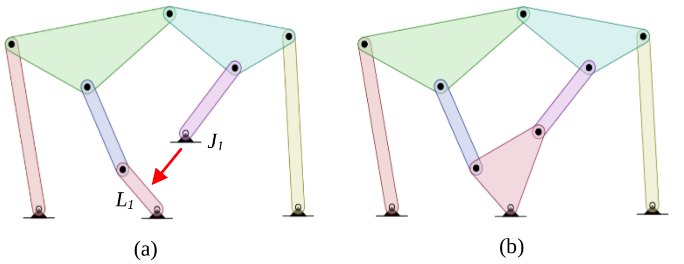


FIGURE 5: A double butterfly linkage is generated after attached ground joint J_1 to link L_1 [13][14].

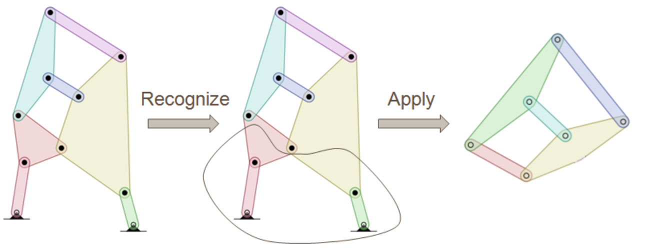


FIGURE 7: An example shows applying a *Remove_Three_Bar* rule to identify a rigid structure.

can also lead to linkages with rigid structures. For example in Fig. 6, the 8 bar linkage contains a 5 bar rigid structure, and it is equivalent to a 4 bar linkage. So this 8 bar linkage is considered as a violation during searching process. These results with rigid structures need to be removed during searching process. On the other hand, for the systematic enumeration of mechanisms, the structural characteristics of rigid structures are valuable to sort in order to ensure that future results do not suffer the same problems. When a rigid mechanism is found, it gives birth to a taboo detection rule, which is automatically generated and check with subsequent mechanism graphs.

The taboo detection process starts with a single rule called *Remove_Three_Bar*. This rule can identify and remove a 3-binary-link chain from the linkage to obtain a rigid structure. This method is based on Gruebler's equation (1). For an L bar 1 DOF linkage with a rigid structure, since it contains a rigid structure, there must be additional structure to drive this rigid component to resolve the whole linkage back to 1 DOF. By introducing new variables to Gruebler's equation (1), the number of links and joints in the driver structure can be found and thus removed to find the rigid structure. The modified equation is as below:

$$3(L-1-L_{remove}) - 2(J_1-J_{remove}) = 0 \qquad (4)$$

The new variables L_{remove} and J_{remove} are the number of the links and joints that can be removed. In order to reduce the DOF from 1 to 0, combinations of L_{remove} and J_{remove} can be found. In this study, the combination of L_{remove} = 3 and J_{remove} =4 is chosen because it can only be represented by a single unique structure which is a 3 binary links chain with 4 joints. Higher values of $\mathrm{L}_{\mathrm{remove}}$ and $\mathrm{J}_{\mathrm{remove}}$ are represented by several topologies. Finding those structures is a tedious process and is not preferred.

By applying this rule, the rigid structures with L-3 number of links can be discovered during L bar linkage generation. For example, the 8 bar linkage in Fig. 7 is generated after applying transformation rule, and since it contains a three binary link chain. This chain is deleted and the rigid structure is obtained. After extracting rigid structures from the linkage, only those that do not contain rigid sub-structures will be saved to the second detection rule set which called *taboo_structures*. Figure 8 shows two 5-bar rigid structures that are captured by applying *Remove_Three_Bar* rule during 8 bar linkage synthesis. For the rigid structure in Fig. 8a, it contains a ternary structure that have been captured and sorted in *taboo_structures* during the early 6-bar linkage searching. So this structure will not be sorted. On the other hand, the structure in fig. 8b does not have any rigid sub-structure, and it will be sorted in the *taboo_structures* rule set for detecting violated results.

P-Joint Substitution

For each valid linkage, revolute joints may be replaced by prismatic joints to increase the diversity of the mechanism space. Although a prismatic joint is considered as a full joint, its angle and the position it is placed can affect the validity of the mobility calculation from Gruebler's equation. For example, recall the four bar linkage from Figure 1 that had 3 P-joints at

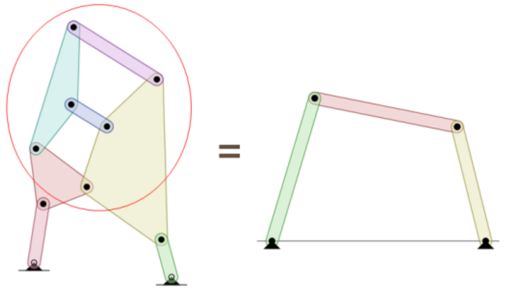


FIGURE 6: An 8 bar linkage contain a 5 bar rigid structure equivalents to a 4 bar linkage [15].

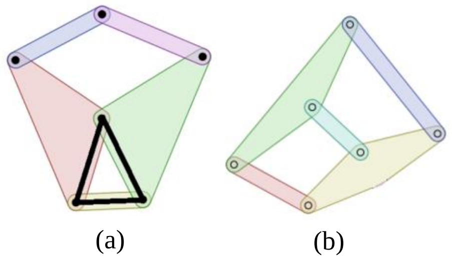
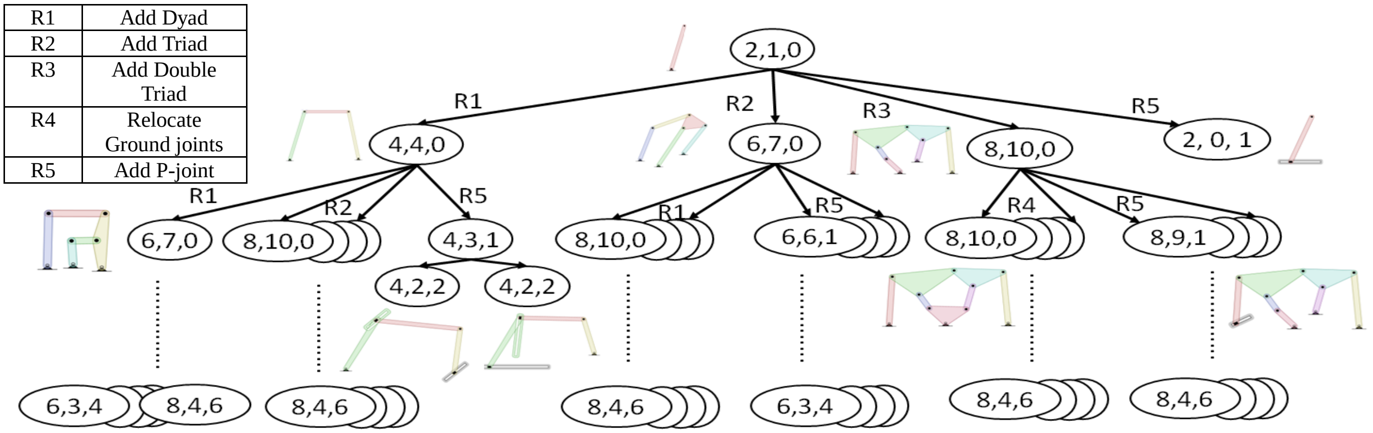


FIGURE 8: Two rigid structures are obtained from 8 bar linkage synthesis.



R1	Add Dyad
R2	Add Triad
R3	Add Double Triad
R4	Relocate Ground joints
R5	Add P-joint

FIGURE 9: A topologies tree search for 2, 4, 6 and 8 bar linkages.

the same angle and thus 2 DOF. In this research, random non-equal angle is assigned to each P-joint to prevent parallel P-joint angle issue.

Even if the angles are different, the mobility of a linkage with P-joints may not conform to Gruebler's equation. To prevent such problems, it is important to properly identify locations to substitute P-joints for R-joints by using two constraints.

The first constraint is the number of R-joints in each loop should not be less than 2 [16]. When placing a P-joint to a recognized R-joint, we need to search all the loops that contain this same recognized joint in the linkage. If one of these loops has less than 2 R-joints after placing the P-joint, like a 4-bars loop in Fig. 1, the R-joint in the loop cannot rotate. The second constraint is that two binary links with only P-joints cannot be connected together. This constraint prevents having a dyad comprised only of P-joints. This leads to an independent sliding in the binary links as well as the prevention of rotation in the neighboring links.

Tree of Topologies

Following this explanation of the graph grammar rules, the rules can now be applied into the search process. Breadth-first tree search (BFS) and depth-first tree search (DFS) methods were tested by using graph grammar rules to synthesis linkages in the early of study. It showed that for the same amount of results, BFS is more adapted to the rules we construct while it generates less isomorphic graphs and finish the searching process faster. So breadth first tree search method is used to populate the states. The procedure of generating 4, 6 and 8 bar linkages with R- and P-joints is shown in Fig. 9. A pendulum with 2 links 1 R-joint and 0 P-joint (#L=2, #R=1, #P=0) is placed in the first level as a seed of the tree because it is the most simple 1 DOF R-joint linkage. The seed is recognized by rules R1, R2, R3 and R5. After applying these rules to the seed, four children graphs are generated in second level. These are the typical four-bar linkage (#L=4, #R=4, #P=0), a six-bar linkage (#L=6, #R=7, #P=0), an eight-bar linkage (#L=8, #R=10, #P=0) and a two-bar linkage with a single P-joint (#L=2, #R=0, #P=1), and the search process in second level is finished This recognize and apply process repeats to generate new levels of candidates until no rules can be recognized, then the tree will contain all the topologies.

After a candidate is created, it needs to be examined by the taboo detection rules. If a rigid structure is found, the candidate will be discarded. If the rigid structure is a new structure and without rigid sub-structure, it will be sorted in *taboo_sturcture* rules set. When this candidate passes taboo structure detection, it will be compared to the topologies that have already been found in the tree. This breadth first search is augmented to detect isomorphism throughout the tree. This candidate will be sorted only if it is the unique result.

The bottom of the tree contains results that have the maximum number of P-joints and no more P-joint can be added. Also, these results are originally generated from R-joints only linkage. Hence, when the tree search process stops, 4,6 and 8 bar linkages with only R-joint or R- and P-joints can be obtain from the tree.

RESULTS

This synthesis method was applied to search all 1 DOF linkage with different number of links and joint types, and the results are compared with other researchers. As is done in the related work, this topological comparison often focuses only on the graph connectivity and no distinction is made as to what link is ground and what joint is the prime mover. Therefore, in the grammar rules and tree search performed here, the local labels *ground* and *input* will be removed. It is important, however, to consider the effects of ground when populating the space for defining distinct kinematic behaviors. For example, the reader is likely aware that six-bar revolute joint mechanisms come in five varieties if ground is considered (Watt 1 & 2, and Stephenson 1, 2 & 3) but just two if ground is not (Watt and Stephenson). In the results below, the six-bar is shown to have only two unique in the following results in order to establish correctness with the related work. However, it is possible with the rules to create the expanded ground-sensitive configurations as well.

Mechanism with only Revolute Joints

The results for 6, 8, 10 and 12 bars 1 DOF linkages are in Tab. 2. For 6, 8, and 10 bar linkages, the results match other authors' findings [1] [2] [3] which are well established. For 12-bar linkage synthesis, 6856 topologies are found, and the amount is the same as Tuttle's [1] and Lee & Yoon's[3] studies. Each of these 12-bar linkages does not contain any rigid structure due to the examination of rigid structure. In Sunkari and Schmidt's study[4], they infer the reason for why Hwang and Hwang having larger results is because the degeneracy testing algorithm they use does not remove all the infeasible solutions. This proves that our dyadic synthesis method correctly spans the design space for mechanisms with only revolute joints. The results for 14-bar linkage is different from Tuttle's [1] and Lee & Yoon's[3] studies. Also, the validity of each result from their studies cannot be tested because there is no available access to those linkages. In this study, the total number of 14-bar linkages was not obtained due to a prohibitively long search process. Our degenerate testing algorithm is time-consuming, but appears to be creating valid 14-bar linkages so far.

This synthesis method is also able to create linkages like 16- or 18-bar mechanisms. Again larger rigid structures may exist, but our detection of rigid structures will ensure that only feasible solutions are presented. While the entire space of solutions may not be explicitly create (e.g. each saved to a computer file), any number of valid solutions can be created.

Mechanisms with Revolute and Prismatic Joints

When replacing R-joints with P-joints in a linkage, the grammar rule is based on the two constraints mentioned in section 4.5. The results for enumerating the size of the space are shown in Tab. 3. All of the 50 6-bar topologies involving combinations of R- and P-joints have been simulated in order to prove their 1-DOF validity. Furthermore, none of them have links that move independently of the prime mover. This result is different than Sardain's study. The reason for that is because the P-joint substitution rule can be applied to the linkage that primary components and secondary components cannot be clearly defined. For example, the dimension of each link in a Stevenson-II 6 bar 4 P-joints linkage (Fig. 10) depends on others. So Sardain's method is not suitable in this situation.

TABLE 2: Results for 6, 8, 10 and 12 bar R-joints linkage compare with other authors' works

	Results	Tuttle	Lee & Yoon	Hwang & Hwang
6- bar	2	2	2	2
8-bar	16	16	16	16
10-bar	230	230	230	230
12-bar	6856	6856	6856	6862
14-bar		318123	275255	

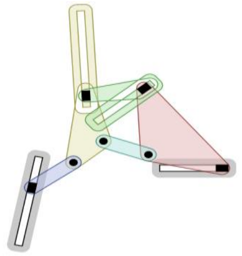


FIGURE 10: A Stevenson-II linkage with 4 P-joints [17].

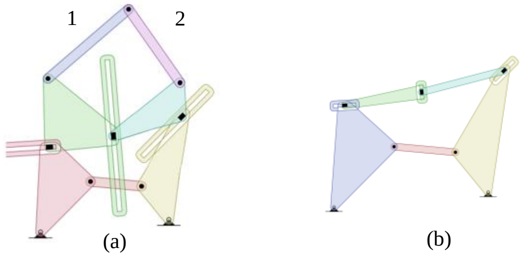


FIGURE 11: An 8 bar linkage contains a PPP chain.

Beyond the 6-bar linkages, values are shown for 8, 10, and 12-bar R and P mechanisms. Unfortunately, at the time of writing, it was realized that errors occur in some of these generated mechanisms. Therefore, the reported numbers that are followed by an asterix (*) are upper-bounds instead of actual values. This is a result of a higher-order constraint that was not captured by the rule and has not been explicitly defined in the literature. For example, the topology in Fig. 11a is an 8-bar linkage with three consecutive P-joints. The positions of these 3 P-joints do not violate the previous rules, but still the linkage has 2 DOF. The dyad (link 1 and 2) in the linkage does not affect the movement of this linkage. After removing this dyad, a new 6 bar linkage is created, and a PPP dyad exists in this linkage which is considered as a violation in Fig. 11b. The new challenge arises to include this as a constraint in the P-joint substitution rule to reduce these numbers to the actual set of feasible R and P mechanism. Part of the difficulty is allowing valid PPP chains such as shown in Fig. 11.

TABLE 3: Results for 6, 8, and 10 bar linkages with P-joints.

		2bars	4bars	6bars	8bars	10bars
Number of P-joints	1	1	1	6	84	2307
	2		2	16	345	12925
	3			19	780*	43883*
	4			9	1083*	97138*
	5				682*	
	6				140*	

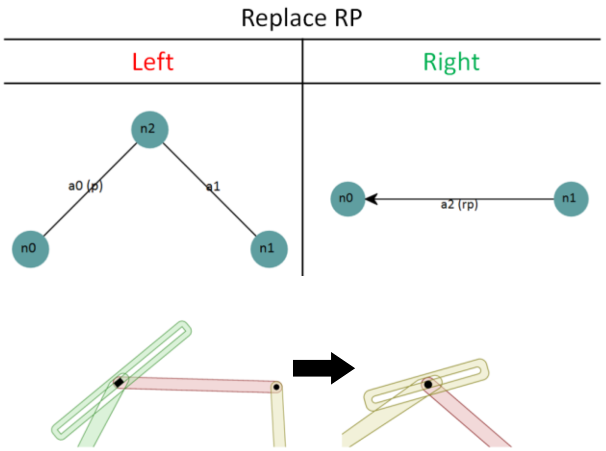


FIGURE12: A *Replace_RP_Joint* rule with its application.

5.3 Mechanism with Revolute and Pin-in-Slot Joints

After having the results with P-joints, we can introduce RP-joints to better represent real mechanisms. This is opposed to Tsai's approach where the RP-joint is a special case of collocating an R-joint at the same coordinates as a P-joint. However, this neglects the fact that the number of nodes will no longer correspond to the number of rigid links. Our goal is to include mechanisms like the scotch-yoke (with L = 3) and various other odd-numbered link mechanisms.

The approach to generate an RP-joint in the graph is to search for binary links with one R-joint and one P-joint. The application of the rule deletes the binary link and replaces the joints with a single RP-joint. The rule is illustrated in Fig 12. Since we have some violations in P-joints results of Tab. 3, we only show results for linkages with R- and RP-joints and the results we have are still valid. The reason for that is when generating certain number of RP-joint, we need the select the linkage that has the same number of binary that contains R and P-joints. So this can avoid selecting a linkage that contains PPP chain.

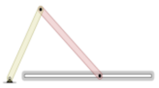
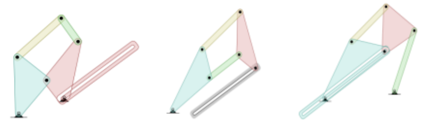
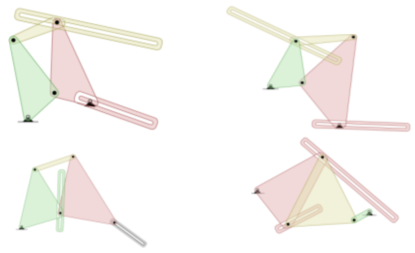
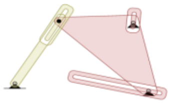
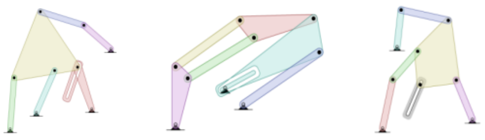
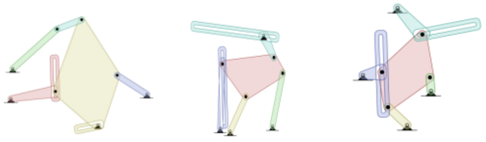
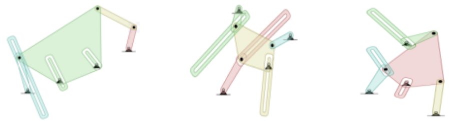
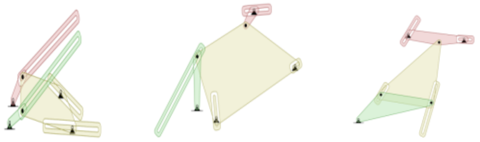
During isomorphism comparison, the direction of the arc representing an RP-joint is excluded since we are only concerned with the position of the RP-joint in the graph. The number of topologies for 3 to 7 bar linkage comprised R- and RP-joints is in Tab. 4.

CONCLUSIONS

This study established a new graph representation for linkages based on Tsai's study. The addition of an RP-joint is introduced to the graph to represent a linkage that has pin-in-slot joint. By applying a graph based tree search, a valid design space for linkages comprised with R-, P- and RP-joints can be explored. Generation, transformation and detection grammar rules are constructed and are used to generate R-joints links.

Generation rules can add components to a linkage without changing its mobility. When a linkage contains multiple ground joints, the transformation rule can detach one of the ground joints and relocate it to another link. After that, rigid

TABLE 4: Results for RP-joints linkages

	Total	Examples
Generated from 4 bar linkage		
3bar- 1 RP	1	
Generated from 6 bar linkages		
5bar- 1 RP	3	
4bar- 2 RP	4	
3bar- 3 RP	1	
Generated from 8 bar linkages		
7bar- 1 RP	32	
6bar- 2 RP	41	
5bar- 3 RP	20	
4bar- 4 RP	5	

structures may be detected within the linkage. So, a special rule checks the validity of the linkage next. The correctness of this method is proven by having the same results to other author's works.

After R-joint linkages are constructed, P-joints can be added to the linkage by replacing R-joints. This substitution is based on the two constraints as described in section 4.5. Although violations exist starting at 8 bars level, the results under 6 bar level are still valid.

For RP-joint generation, a binary link with R- and P-joint is replaced by a RP-joint. Since violation happens by applying P-joint, we search for linkages that only have R- and RP-joint. So that it can prevent using violated results. The results for the enumerations of the space of RP-joints topology is an innovation presented in this paper.

Now a large space of possible configurations has been produced. One can use this repository to search for topologies that would best serve the design requirements for a particular mechanism design. Although, we limited the process to 12-bars and fewer in this study, many aspects of the graph rules could be extended to mechanisms with even more rigid links.

REFERENCE

[1] Tuttle, E., 1996, "Generation of planar kinematic chains," Mechanism and Machine Theory, **31**(6), pp. 729–748.

[2] Hwang, W. M., and Hwang, Y.W., 1991, "An algorithm for the detection of degenerate kinematic chains," Mathematical and Computer Modelling, **15**(11), pp. 9–15.

[3] Lee, H. J., and Yoon, Y.-S., 1994, "Automatic Method for Enumeration of Complete Set of Kinematic Chains," JSME International Journal. Ser. C, Dyn. Control. Robot. Des. Manuf., **37**(4), pp. 812–818.

[4] Sunkari, R. P., and Schmidt, L. C., 2006, "Structural synthesis of planar kinematic chains by adapting a Mckay-type algorithm," Mechanism and Machine Theory, **41**(9), pp. 1021–1030.

[5] Tsai, L. W., 2010, Mechanism Design: Enumeration of Kinematic Structures According to Function, CRC Press.

[6] Woo, L. S., 1967, "Type Synthesis of Plane Linkages," Journal of Engineering for Industry, 89(1), p. 159.

[7] Sardain, P., 1997, "Linkage synthesis: Topology selection fixed by dimensional constraints, study of an example," Mechanism and Machine Theory, **32**(1), pp. 91–102.

[8] Radhakrishnan, P., and Campbell, M. I., 2010, "A Graph Grammar Based Scheme for Generating and Evaluating Planar Mechanisms," Design Computing and Cognition, pp. 653–669.

[9] Campbell, M. I., 2015, "PMKS: Planar Mechanism Kinematic Simulator". URL purl.org/pmks/. [Accessed: 01-Jan-2015].

[10] Huang, W., and Campbell, M. I., 2015, "Planar Mechanism Kinematic Simulater:3 Bar with 1 RP(1)". URL purl.org/pmks/MR1.

[11] Huang, W., and Campbell, M. I., 2015, "Planar Mechanism Kinematic Simulater:3 Bar with 1 RP(2)". URL purl.org/pmks/MR2.

[12] Campbell, M., 2009, "A Graph Grammar Methodology for Generative Systems," Methodology, pp. 1–25.

[13] Huang, W., and Campbell, M. I., 2015, "Planar Mechanism Kinematic Simulater: 8 Bar with 10 R". URL purl.org/pmks/MR4a.

[14] Huang, W., and Campbell, M. I., 2015, "Planar Mechanism Kinematic Simulater: Double Butterfly". URL purl.org/pmks/MR4b.

[15] Huang, W., and Campbell, M. I., 2015, "Planar Mechanism Kinematic Simulater: 8 Bar with RigidStructure". URL purl.org/pmks/MR5.

[16] Pucheta, M., and Cardona, A., 2005, "Type synthesis of planar linkage mechanisms with rotoidal and prismatic joints," Mecánica Comput., **XXIV**, pp. 2703–2730.

[17] Huang, W., and Campbell, M. I., 2015, "Planar Mechanism Kinematic Simulater: Stevenson-II with 4 P". URL purl.org/pmks/MR6.