Engineering with Computers (1999) 15: 303–314
© 1999 Springer-Verlag London Limited

Engineering with Computers

Isomorphic Representations and Well-Formedness of Engineering Systems

O. Shai[1] and K. Preiss[2]

[1]Department of Solid Mechanics, Materials and Structures, Tel Aviv University, Tel Aviv, Israel; [2]Department of Mechanical Engineering and School of Management, Ben Gurion University, Beer Sheva, Israel

Abstract. *When representing the elements of different engineering systems as vertices and edges of a mathematical graph, the well-formedness of the topology of the graph, and hence the topology of the engineering system, can be explicitly computed. This enables checking the well-formedness of the engineering system before investing the effort needed for a complete analysis. This method is demonstrated in the paper for the following fields: trusses, dynamic mass-spring-damper oscillator systems and planetary gear systems. The approach facilitates achieving rapid design, correct first time, which is an important aim of modern design computation.*

Keywords. Dynamic systems; Graph theory; Isomorphic representations; Planetary gear systems; Trusses; Well-formedness

1. Introduction

A number of modern studies of requirements for manufacturing [1,2] have analysed the new requirements that follow from global competitive pressures. A significant observation is that global competition requires greatly increasing speed when bringing new products to market. Many companies are attempting to speed up work processes, including design, by a factor of ten – from a year to a month, from a month to three days [3]. This in turn has increased the pressure that design work be achieved quickly. One increasingly hears the sentiment that a design must be 'right first time'. To achieve an environment where design is both fast and correct the first time it is made, it is useful to revisit the computation methods used for design, to see where improvements may be made. Systematic analysis of the well-formedness of the engineering system being computed, as presented here, is a contribution to such an improvement in speed, or as it is sometimes termed, cycle time reduction.

The approach adopted is similar to that used when evaluating formulae or expressions in any mathematical system. The work is divided into two parts. First, the expression or formula is checked to see if the arrangement of symbols is well-formed. If it is not a Well-Formed-Formula (WFF), it cannot be solved. If it is a WFF, the evaluation procedure can be invoked.

For example, imagine the following string of symbols in an 'arithmetic' expression

$$2 + -3.!4.6 * a = 4,.2 + 3.6.2.1$$

This syntax of this expression is not a well-formed-formula and it cannot be evaluated.

A similar situation holds when wanting to compute a solution for a given engineering system. In order to apply the equations of a computation method the system must be well-formed, otherwise the equations will not be applicable. For example, if a structural truss has insufficient members, or if they are distributed incorrectly in the structure, it is in fact a mechanism, the force in some members will always be zero, and the methods of computing forces in the members will not be applicable. If a mechanism has too many elements, or if they are incorrectly placed in the mechanism, it will in fact be a structure. Some or all members will then not be able to move, and the methods used for determining movement and velocity in a mechanism will not be applicable. To compute the behavior of an engineering system, the stages are therefore:

1. Evaluate the well-formedness of the system. For example, if it is a truss, determine if it is rigid, and if so whether it is statically determinate or

Correspondence and offprint requests to: Dr O. Shai, Department of Solid Mechanics, Materials and Structures, Tel Aviv University, Tel Aviv 69978, Israel

indeterminate. If it is not rigid, the analysis stops at this point; otherwise,
2. Solve the system using an appropriate method.

Many engineering systems are composed of elements that are connected together. Well known examples are: electronic circuits where resistors, capacitors and other components are connected; a structure where axial force members, beams and plates are connected; a mechanism where rotating and moving elements are connected, and so on. The information as to which elements are connected to which other elements is known as the 'topology' of the system. The topology does not include geometric information such as the dimensions of the elements. For a complete definition of the system one needs both the topological information and the geometric or other properties, such as heat flux or temperature, of the elements and their connections.

The examples shown in this paper are for determining whether the system has a well formed topology. If it does not, it cannot be solved. If it does, there may still be values of geometric or other variables that create singularities in the solution. Well-formedness of the topology is a necessary condition for solubility, but is not sufficient. The necessity but insufficiency of topological well-formedness is analogous to a well-known result for the boundary representation in solid modelling, where similarly topological well-formedness is a necessary but insufficient condition. Good modern boundary representation solid modelers make extensive use of that fact, by separating the computation of topological consistency from geometric consistency, in modular software. The usual abbreviation for 'well formed formula' is WFF. For 'well formed topology' we will use the abbreviation WFT.

Systematic evaluation of the well-formedness of a system is conveniently achieved using mathematical representations which are isomorphic to the system. Isomorphism is discussed in the next section. The isomorphic representations used in the Embedded Knowledge Project [4] which formed the basis of this paper were matroid theory and graph theory. In the interests of brevity, the examples in this paper show only the graph theory representation.

2. Isomorphic Representations

A mathematical representation is isomorphic to an engineering system if every element in the engineering system corresponds to an element in the mathematical representation, and *vice versa*. A mathematical graph is composed of vertices connected by elements, and this is a mathematical representation that is conveniently used as a representation of an engineering system.

The usual practice for analysing an engineering system is to choose variables which represent the behavior of the system – for instance, forces or deflections in a structure, or velocities or movements in a mechanism, or temperatures or heat fluxes in a heat transfer problem, and so on. The behavior of the system is then described by a set of equations written in terms of material properties and the chosen behavioural variables. These equations express the behaviour of the system, but are not isomorphic to the elements of the system itself. If the engineering system is not well-formed, the mathematical manipulations lead to impossible situations, such as a division by zero, and then one knows there had been a problem in the definition of the engineering system.

The approach suggested here is to choose explicit variables for the system which are isomorphic to the system elements, then to evaluate the well-formedness by using mathematical properties embedded in that representation. For most engineering systems, a necessary condition for well-formedness is that the topology of the system is well-formed. By using an isomorphic representation one can check the well-formedness of the topology and be sure of its validity before proceeding further with the analysis. The full analysis can be achieved directly from the isomorphic representation, but the methods for this are not presented in this paper. Some information on analysis methods based on graph theory is available elsewhere [5,6]; papers giving further details of analysis methods are in preparation.

Application of theorems and algorithms published in the literature of graph theory over the past few decades make graph theory even more useful than before for dealing with physical engineering systems. The use of graph theory in engineering has been known for decades, since the work of Kirchhōff in 1847, who applied graph theory to electrical circuits. Graph theory has since been applied to many fields of engineering. In 1962, Kron [7,8] used graph theory to analyse elastic networks. He used the analogy between electrical networks and elastic systems and developed the method called 'diakoptics'. Fenves and Branin [9] published a method based on graphs and networks for the formulation of structural analysis. Since then, other works have been reported, for example Lind [10], Kaveh [11], Preiss and Shai [6], and many others. Work published at Waterloo University [12] shows that graph theory is useful for solving dynamic systems. Graph theory is being applied in many other fields, such as the design and analysis of integrated microelectronic circuits, machine theory and operations research, where the main use is to analyse or find a solution to a specific engineering problem. The approach reported in this paper is different. We present an approach whose purpose is to check the validity of the engineering system before commencing the analysis. Graph theoretic analysis can usually be performed in much less time than solving for the full behaviour of the system, and if a topological inconsistency is found before progressing to full analysis, time and effort can be saved. This is thus a process of checking well-formedness of engineering systems. Matroid theory, which can be thought of as a generalisation of graph theory, contributes to some of the algorithms used. When computation is performed on the isomorphic graph representations, using the methods and algorithms embedded in the graph theory representation and derived from matroid theory, the computational effort is reduced.

In a graph there are only two types of elements: vertices and edges. The first question, therefore, when choosing the representation, is to decide which type of element in the engineering system should be associated with vertices or with edges. Table 1 shows the relationship between the vertices and edges of a graph, and the elements of three engineering systems; structural trusses, dynamic mass-spring-damper systems, and planetary gear systems. The table shows, for each such engineering system, the names of the elements, the connectivity between them, and the correspondence, which is unique, of these to the elements of a graph.

3. Examples

The three engineering systems shown in Table 1 are usually considered as belonging to different fields. Although combined analyses can be made, for instance of the combined effects of static and dynamic forces in a planetary gear system subject to inertial accelerations, the fields of structural statics, machine theory and dynamics are usually considered separately, with separate analysis techniques. However, each of these apparently distinct systems can be represented as a graph that is isomorphic to the engineering system. Assembling the graph does not require specialist knowledge, because it is done algorithmically. Checking the well-formedness of the topology is then done, also algorithmically, enabling automation of this computation, and not requiring specialist human expertise. If the system is not well-formed it cannot be solved, is usually in practice a defective system, and the work stops at this point. If it has a well-formed topology, assembling the solution equations and solving the system can then also be done algorithmically on the graph representation.

In other words, given a truss, we apply a mathematically rigorous and complete check to ensure that it is rigid before applying the solution techniques. So too with the planetary gear and dynamic systems, and the approach has been applied to other engineering systems [4]. In all cases checking the well-formedness of the engineering system is done on the mathematical representation which is isomorphic to the engineering system.

Well-formedness is checked using rules for the topology, which are rules for the syntax of the graph. Some rules are universal for graphs in gen-

Table 1.

Engineering system	Truss	Dynamic mass	Planetary gears
The engineering elements	Rods. External Forces. Supports.	Masses. Springs. Dampers. External forces.	Gear wheels. Rotation links
Connections between the engineering elements	Pinned-joints connect between the rods and connect rods to supports.	Pinned-joints	Gear connections. Turning connections.
Correspondence between the engineering elements and the graph elements	Rod –> edge. Pinned-joint–> vertex.	Dynamic element –> edge. Pinned-joint –> vertex.	Links —> vertices. Gear and turning connections –> edges.

eral, and others are specific to the type of engineering system being dealt with. The topological rules are embedded in the structure of the graph. Checking the well-formedness of an engineering system is therefore a process of checking that there is no contradiction between the structure of the graph, which is an isomorphic representation of the engineering system, and the rules and theorems that are embedded in that representation. Although graph theory is used also for the next step, analysis of the engineering system, in the interest of brevity that is not explained in this paper which is limited to analysis of the well-formedness of the topology. Examples of engineering systems dealt with in the paper and their corresponding graphs are shown in Fig. 1.

4. Checking the Topological Well-Formedness of a Truss

The word 'rigidity' is used here when referring to the truss structure without its supports, and the word 'stability' is used for the truss structure with its supports.

For a stable truss the rigidity matrix contains no singularity. A singularity can exist if the truss topology is not well-formed, or if it is well-formed but exhibits geometric singularities. The well-formedness dealt with here is the topological well-formedness, not geometric well-formedness. As shown below, the same topological well-formedness algorithm is applicable to the truss alone, when dealing with the well-formedness of the structure alone, or to the truss and the reactions together, when dealing with the whole structural system.

4.1. The Graph Representation of the Truss

The graph that represents a truss can be obtained algorithmically by executing the following steps:

1. Assign a vertex to every pinned joint of the truss.
2. Assign an edge to every rod of the truss, its end vertices corresponding to those pinned joints that connect the rod to the truss.
3. Create two extra vertices called 'X' and 'Y' and connect them with an edge.
4. For every pinned support create two edges connecting the vertex corresponding to the support, with the X and Y vertices.
5. For every mobile support create an edge connecting the corresponding vertex with X (or Y) if the support is mobile on the vertical (or horizontal) plane, respectively. If the support is mobile on some inclined plane, create an

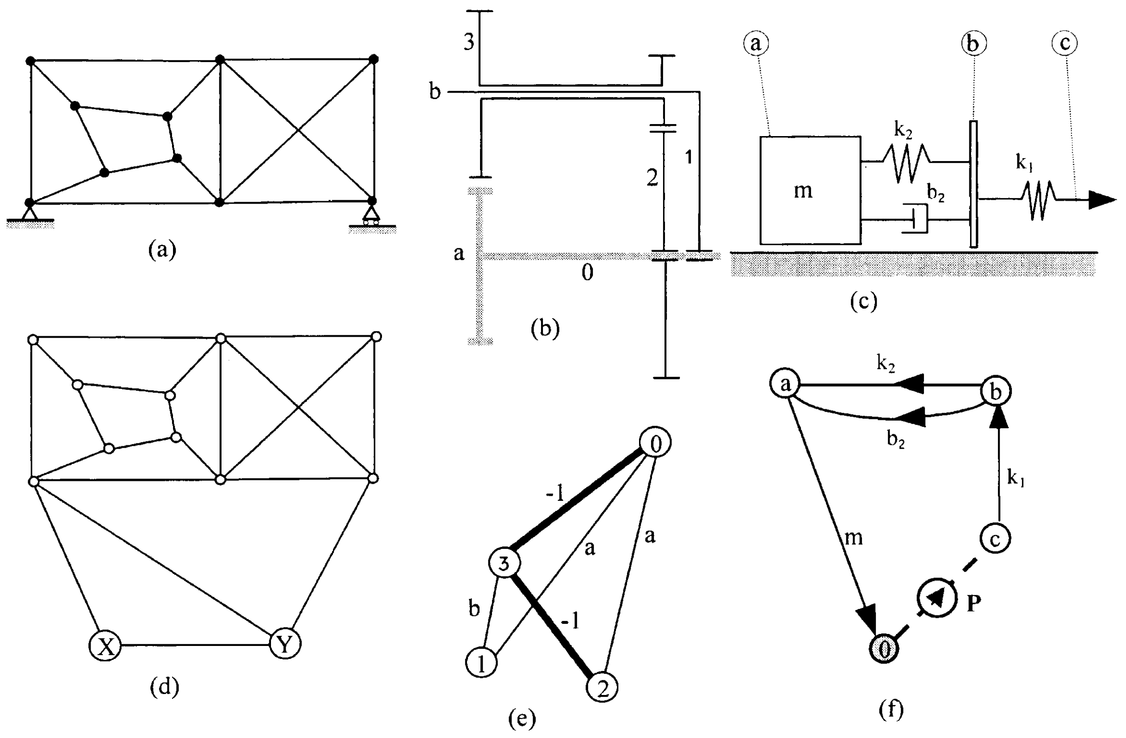


Fig. 1. Examples of engineering systems and the corresponding graphs discussed in the paper.

additional vertex and connect it using three edges, to the vertices named X and Y and to the vertex corresponding to the support.

Note that edges representing applied loads do not appear. These edges do not affect the topological consistency of the graph, and the loads do not affect the rigidity of the truss or the stability of the truss on its supports.

Adding the two additional vertices X and Y and those which correspond to the inclined planes as described above (steps 3 to 5) enables checking whether the graph of the truss and its reactions is well-formed, meaning that the truss and its supports have a well-formed topology. If one is interested in the well-formedness of the truss structure itself, without the reactions, the links for the reactions (steps 3 to 5) are omitted.

Figure 2(b) shows the graph isomorphic to the truss of Fig. 2(a). Since pinned-joint ‘a’ is connected to a fixed support, two edges, one for each coordinate, appear in the graph, and for the mobile support ‘d’ there is only one edge ‘dx’.

Denoting $e(G)$ and $v(G)$ as the number of edges and vertices respectively in the graph G, trusses are divided into three groups:

(a) if $e(G) < 2v(G)-3$ then the truss is a priori not rigid, and the computation stops, else
(b) if $e(G) = 2v(G)-3$ then the truss, if rigid, is determinate, else
(c) if $e(G) > 2v(G)-3$ then the truss, if rigid, is indeterminate.

4.2. The Theorems Embedded in the Graph Representation

Most of the published literature on the subject of rigidity of trusses deals with determinate trusses. There then exists a fixed relation between the number of rods e and pinned-joints v, or in the terminology of the graph, $e = 2^* \ v - 3$. Maxwell [13]

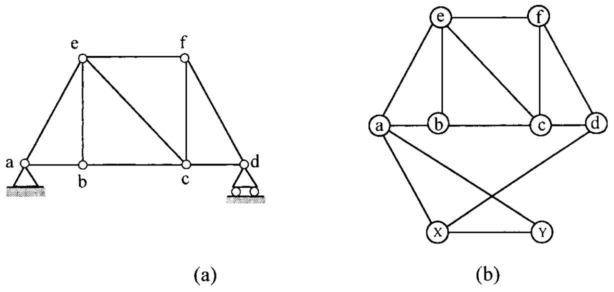


Fig. 2. An example of a truss (a) and the graph that represents it (b).

proved that if the relation $e' \leq 2^*v' - 3$ holds for every sub-graph of G, then the corresponding determinate truss is rigid. About 100 years later, Laman [14] proved that this condition is not only necessary, but also sufficient.

The connection between the rigidity of determinate and indeterminate trusses is established by the following theorems. The proof of Theorem 4.2 requires graph theory and matroid theory.

Theorem 4.1 Let G be a graph that corresponds to an indeterminate truss. Then G is rigid if and only if (iff) there exists in G a connected subgraph G' which includes all the vertices of G, corresponds to a determinate truss, and is rigid.

Proof If G' is a determinate truss and is rigid, adding edges (rods) does not destroy the property of rigidity. The inverse connection between G and G' follows directly from the definition of an indeterminate truss. Suppose that in G there are k redundant rods, G then is said to have a redundancy of k. When deleting those k edges from the graph of G, the truss represented by G' remains determinate.

For a determinate truss, the necessary and sufficient condition for checking whether the truss has a well-formed topology is shown in Theorem 4.2.

Theorem 4.2 [15] A determinate truss is rigid if and only if (iff) when doubling each edge in turn in the corresponding graph, all the edges can be covered by two edge disjoint spanning trees.

From this theorem one can derive the following algorithm for checking the well-formedness of the topology of determinate trusses.

4.3. Method for Checking the Validity of the Graph of a Statically Determinate Truss

The following algorithm is based on Theorem 4.2.

Algorithm 3.1 Checking the well-formedness of the graph of a determinate truss

(1) Build the graph corresponding to the truss, as was explained in Section 4.1.
(2) For every edge in the graph do:
double the edge and search for two edge disjoint spanning trees by using known algorithms [16].
(3) If step 2 is successful for every edge in the graph, then the graph has a well-formed topology, otherwise not.

For example, Fig. 3 shows a truss (Fig. 3(a)) and

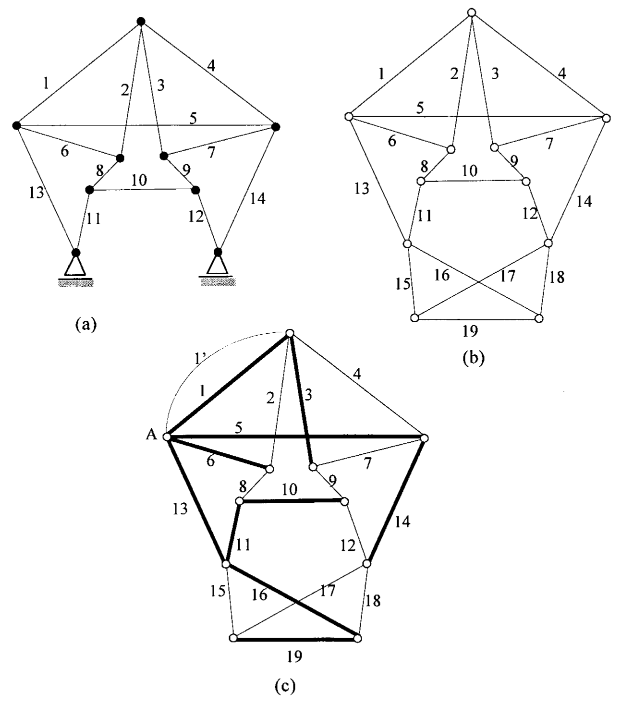


Fig. 3. Example of a proof that a determinate truss (a) is stable. (a) The truss, (b) the corresponding graph, (c) two edge disjoint spanning trees when doubling edge ‘l’.

its corresponding graph (Fig. 3(b)). It can be proved to be stable, since when doubling each edge in turn, it has two edge disjoint spanning trees. Figure 3(c) shows two edge disjoint spanning trees covering the graph when edge ‘l’ is doubled.

4.4. The Applicability of the Method to Complex Trusses

This approach enables the use of algorithms and theorems that have been published over the last few decades in the computer science literature, reducing computation complexity. For example, in Section 3.2 which uses Laman theory, computation complexity is reduced from exponential to polynomial. This can be achieved also by using other theorems and techniques of graph theory. The Tutte theorem [17] can be used to determine whether two edge disjoint spanning trees exist in a graph. This theorem states that there are two edge disjoint spanning trees if and only if (iff) for every decomposition of the vertices of G into m nonempty classes, there are at least $2^*(m-1)$ edges that connect vertices from different classes. The truss on Fig. 4 has a partition into eight classes while there are only 13 edges between them, which proves that this truss is not rigid.

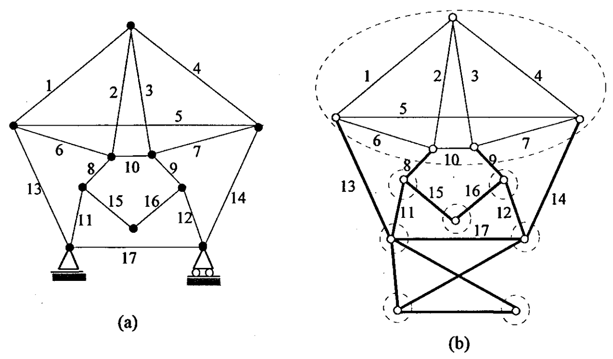


Fig. 4. Example of a proof that a determinate truss (a) is not stable. (a) The truss, (b) the corresponding graph and its partition.

5. Checking the Validity of Dynamic Systems

A similar process as above for trusses can be applied to a dynamic mass-spring-damper oscillator system. In this section, it will be shown that one can find a contradiction in the topological structure of a dynamic system with given initial conditions, by analyzing only the syntax of its graph. Given a dynamic system with initial conditions, there can be a solution only if its graph has a well-formed-topology according to the well-formedness rules shown below.

The approach described below is a part of a more general approach taken in the Embedded Knowledge Project [18], in which for every edge in the graph there is a flow and every vertex has a potential. In dynamic systems the flow in the edge corresponds to the force in the physical element, and the potential at the vertex corresponds to the velocity of the joint.

5.1. The Graph Representation of the Dynamic System

The first step is to build the isomorphic graph representation of the dynamic system, as in the above section dealing with trusses. This is achieved by executing the following steps:

1. Assign a vertex to every junction that can have an independent velocity. Create an additional vertex corresponding to the inertial reference frame, called ‘the reference vertex’, denoted by ‘O’ and labeled with a gray color.
2. Assign an edge to every element of the dynamic system. If the element is a spring or a damper, the end vertices of the spring-edge correspond to the end junctions of the element. If the element is a mass, one end vertex is always the reference

vertex 'O' and the other corresponds to the junction of the mass and the other elements of the dynamic system.

3. For every external force there is a corresponding directed bold dashed edge from the reference vertex 'O' to the junction where the external force is applied. Consistent with the notation in the paper, this flow source edge corresponds to the flow (or force) in the edge that acts on the junction. When a plus sign is associated with the flow, it means that the force acts on the junction in the direction of the edge; a minus sign means it acts in the opposite direction.
4. For every imposed external velocity there is a corresponding directed bold solid edge, where the potential difference (or relative velocity) is positive if it is in the direction of the edge, otherwise it is negative.

With this notation, it is possible to include the initial conditions in the isomorphic graph representation, as follows:

5. Every spring with initial tension will be represented by two parallel edges. Based on the superposition principle, one edge will represent the flow source with the value of the initial tension of the spring, and the other will represent the flow (force) change in the spring caused by the changes of the dynamic system. The meaning of the sign is the same as was explained in step 3 for external forces.
6. Every mass with initial velocity will be represented by two serial edges. Based on the superposition principle, one edge represents the source of potential difference with a value equal to the initial velocity value of the mass, and the other represents the change in potential (velocity) with time in the dynamic system. The meaning of the sign is the same as was explained in step 4 for an imposed external velocity.
7. For all the other edges, an arbitrary direction can be given, but maintaining consistency for the meaning of the direction, which is that the flow (force) in the edge is the force that the corresponding dynamic element applies to the junction which corresponds to the head vertex of the edge.

For example, Fig. 5 shows a dynamic system with initial conditions. The edge $\Delta(M2)$ has a plus sign because the initial velocity of mass M2 is in the positive direction of the x coordinate. The edge Pk2 has a plus sign, because the initial force in the spring is compression and the corresponding edge shows the force (or flow) that the spring k2 applies

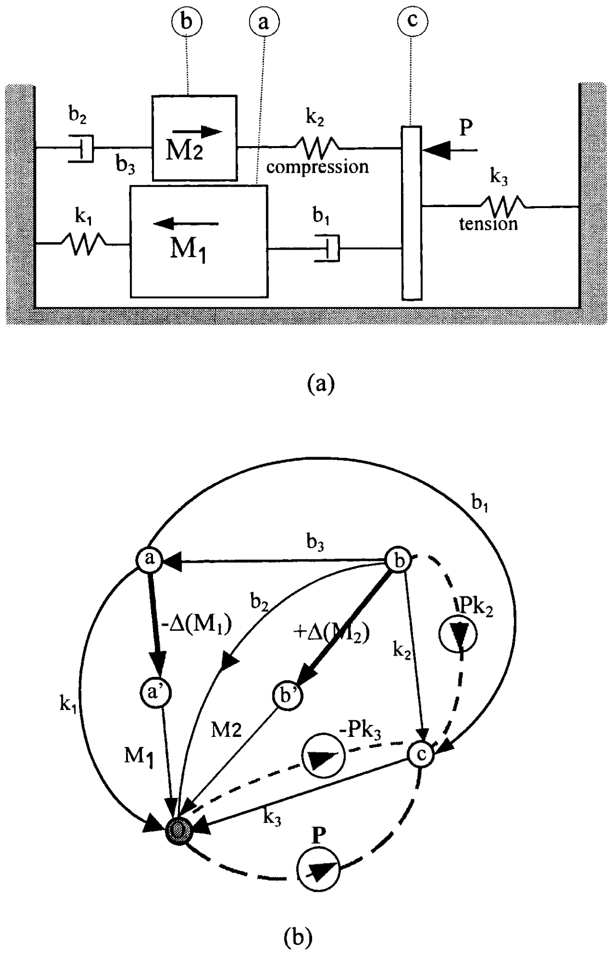


Fig. 5. A dynamic system (a) and its corresponding graph (b).

to mass M2, which is in the negative direction of the x coordinate.

One can see that this representation is easily applied to other complex dynamic engineering systems, for example, a multi-dimensional dynamic system with axial forces. Figure 6 shows a two-dimensional system, where the displacement of spring K3 in the 'x' direction is negligible relative to the displacement in the 'y' direction.

5.2. The Theorems Embedded in the Graph Representation

In this representation there are two theorems:

1. Flow law – the vector sum of flows at any cutset of the graph should be equal to zero.
2. Potential law – the vector sum of potential differences in any circuit should be equal to zero.

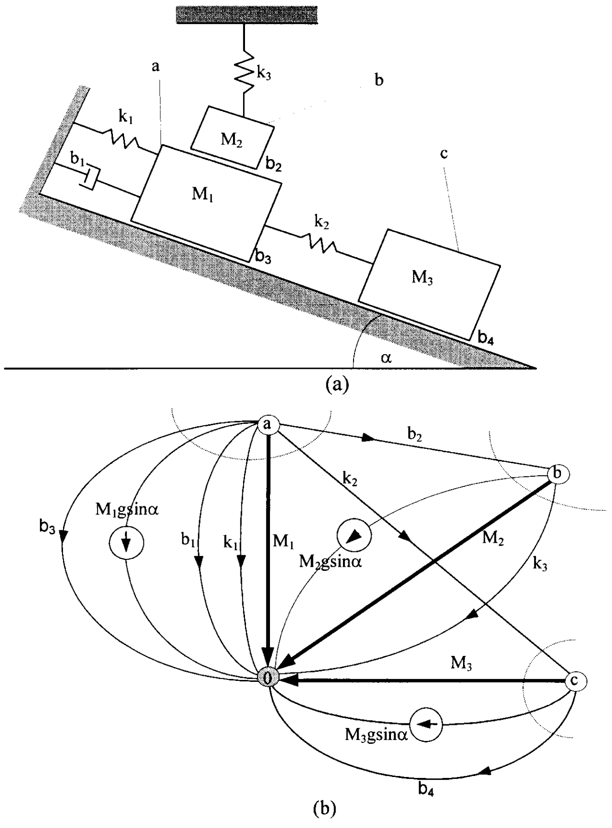


Fig. 6. A two-dimensional dynamic system (a) and its corresponding graph (b).

5.3. The Method for Checking the Validity of the Dynamic System with Initial Conditions

In order that Flow Law and Potential Law will be satisfied, one has to check the following syntax rules:

- *Syntax rule for cut-sets*: there should not be a cut-set of only flow sources (bold dashed edges).
- *Syntax rule for circuits*: there should not be circuits consisting only of potential sources (bold solid edges).

The reason for these restrictions is derived from the property of a source edge. For example, if there were a cutset of only flow source edges, the amount of flow would be independent of the state of the dynamic system and the sum over a cut-set might not be equal to zero, in contradiction with the Flow Law. A similar reason holds for potential source edges.

An example of a dynamic graph representation that does not satisfy the syntax rule for cut-sets is shown in Fig. 7. Figure 7(b) shows that the dynamic graph representation does not satisfy the syntax rule

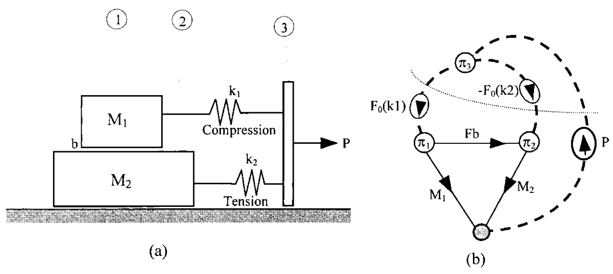


Fig. 7. An example of dynamic system with initial conditions that contradicts the syntax rules of the graph.

for cut-sets, because it has a cut-set with only bold dashed lines. For such a graph, the initial forces around junction 3 might possibly not satisfy the condition of equilibrium of forces.

6. Checking the Validity of a Planetary Gear System

The same general process explained above is now invoked for dealing with planetary gear systems. First, check if the gear system is valid by checking if the system has a well-formed topology. In the truss problem, an efficient method was embedded in the representation for checking the necessary and sufficient conditions for deciding the validity, or rigidity, of the determinate truss. For planetary systems, the necessary conditions follow [19]. These conditions will now be phrased as rules to check if the representation has a Well-Formed-Topology (WFT), following which the embedded theorems in the particular graph, the spanning tree, are used.

6.1. The Graph Representation of the Planetary System

The most important property to be emphasized in this representation is the connection between the system elements, showing how element ‘i’ is connected to element ‘j’. For this reason, every system element will be represented by a vertex, and every connection by an edge. This type of graph is called a ‘line graph’ [4]. The line graph representation is suitable for this purpose, and therefore every rotation rod or gear wheel will be represented by a vertex, and the connection between a pair of links by an edge. There are two types of connections, so there are two types of edges, marked as bold and regular as explained below.

(a) Bold edge (bold line) – by knowing the ratio of the connected gear wheels, one can calculate the ratio between the angular velocities (potentials) of the gear wheels, hence in the terminology of this paper, this edge is a dependent potential source, and for this reason it appears in the graph as a bold line.

(b) Regular edge (solid line) – an edge which represents a turning connection. It will be shown below that the turning edges form a spanning tree.

Other information about the labeled edges and the vertices is added to the representation as follows:

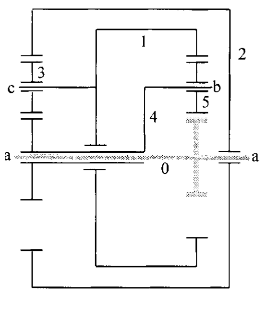
(c) Labelled solid line – every solid line (turning edge) has a label which represents the level, being the location of the rotating connection.

(d) Reference vertices – the distance between each pair of connected gear wheels must be constant all the time, being maintained by a link or planet carrier. This is called in the literature [20] a 'transfer vertex'. In the terminology of this paper, the name 'local reference vertex' is more suitable. In this representation, all the turning edges on one side of the local reference vertex are at the same level, and those on the opposite side of the local reference vertex are at a different level.

(e) Labelled bold line – every bold line (gear edge) has a label which represents the planet carrier (local reference vertex) that maintains the distance between the two gear wheels which correspond to the end vertices of the gear edge. In addition, the bold line has a sign, where the 'plus' (or minus) sign means that the transmission of movement between the two gear wheels is internal (or external).

(f) Labelled gear wheel vertex – every vertex that corresponds to a gear wheel has a label that represents its center level.

Note: Fig. 8(a) is a standard representation in engineering drawing for a gear system.



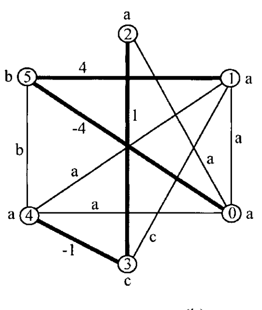


Fig. 8. A planetary mechanism (a) and its line graph representation (b).

6.2. Theorems Embedded in the Graph Representation of the Planetary Gear System

The embedded properties in the graph representation of this problem, given below, are based on Erdman [19], who published a set of necessary conditions which he used for the mechanism synthesis problem. We use them to deduce whether or not the system is topologically infeasible.

Proposition 1 There is no circuit formed exclusively by turning edges.

Explanation 1 Suppose in contradiction to the rule that a circuit of turning edges exists. There would then be, in the chain, a set of pin-connected links. A circuit of sizes 1 or 2 is not feasible. A circuit of size 3 is a triangle which is a locked structure. In a circuit of size 4 or more the rotatability of the links would not be proportional. This contradicts the hypothesis that the system is a kinematic chain.

Proposition 2 All the vertices must be incident to at least one turning edge.

Explanation 2 Every element (link) which is represented as a vertex has at least one element which rotates around it. Between these two elements, there will be a turning edge in the graph representation. There are elements, such as a planet carrier, for which the vertex that represents them is incident to at least two turning edges.

Proposition 3 The subgraph of the turning edges forms a connected subgraph.

Explanation 3 Each connected gear pair should operate with a constant radius or center distance. This distance is maintained by the planet carrier, which is either directly paired to ground or connected to ground through a sequence of turning edges.

Proposition 4 In each fundamental circuit, there is one and only one local reference vertex; all the edges on one side of the local reference vertex are all at the same level, while all the edges on the opposite side of the local reference vertex are at a different level.

Explanation 4 Each gear pair is located on a different turning edge level. Because the distance between the centres of these two gears must be constant, there is one and only one planet carrier (local reference vertex) in the fundamental circuit defined by this gear pair.

In addition, in the graph representation syntax, there are embedded properties, part of which are listed in Table 2. The Gruebler theory referred to in

Table 2. Some of the embedded properties of the line graphs which correspond to planetary gear systems.

	Embedded property	Derived from	Graph theory formulation
1	The subgraph formed by turning edges is a spanning tree.	Propositions 1, 2 and 3.	A subgraph that is connected, with no circuits, and contains all the vertices of the graph, is a spanning tree.
2	Every gear edge forms a fundamental circuit with the spanning tree.	Embedded property 1.	Adding an edge to the spanning tree forms one and only one circuit.
3	$e(T) = v(G) - 1$.	Embedded property 1.	The number of spanning tree vertices is one more than the number of its edges.
4	$g(G) = v(G) - 2$.	From embedded property 1 and Gruebler theory.	
5	$g(G) = e(T) - 1$.	From embedded properties 3 and 4.	

T – the spanning tree.
G – the line graph.
$g(G)$ – the number of gear edges.
$e(T)$ – the number of spanning tree edges.
$v(G)$ – the number of vertices.

the table is well known in theory of machines, and can be found, for instance, in Erdman [19].

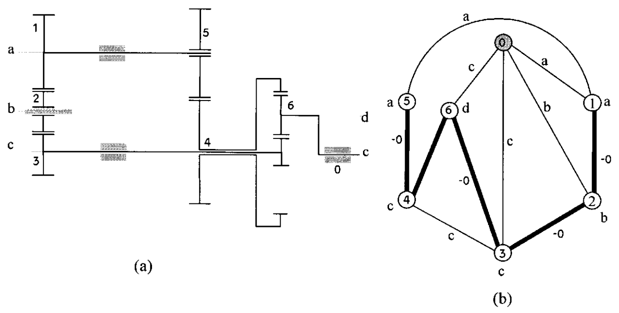
Proposition 5 In each fundamental circuit, the levels of the vertex representing a gear wheel and the turning edge incident to it must be identical.

Explanation 5 The geometric center of a gear wheel and its local center of rotation must coincide.

6.3. A Method for Checking the Validity of the Planetary System

With this representation, checking the validity of the system becomes a problem of checking whether there is a contradiction between the domain knowledge (in this case, the embedded properties and propositions) and the representation of the given system. For example, the computer program using this representation [21,22] found that the system in Fig. 9 is not valid, and explained why. In addition, it is possible to arrange that the computer program advises the designer what to change in the gear kinematic chain, so that it would be valid.

By this approach, one can conclude whether a working gear system is a well-formed planetary gear system or not. For example, according to proposition 5, the gear system in Fig. 10(a) is not a well-formed planetary gear system, since the turning edge that enters vertex ‘l’ is labeled ‘a’ while the gear vertex ‘l’ is labeled ‘c’.



```
The system is not valid because there is a contradiction with
proposition 4. In circuits {6,0,3} and {6,0,3,4} there is no local
reference vertex. Moreover, the level of the turning edge (06) is 'c'
while the level of vertex 6 is 'd', which contradicts proposition 5.

The explanation to the user is: The connection between wheels 6 and 3
is not legal because the distance between their centers is zero.
The same problem occurs with the connection between wheels 6 and 4.
```

(c)

Fig. 9. Example of topological analysis of a planetary gear system, with the computer program output shown.

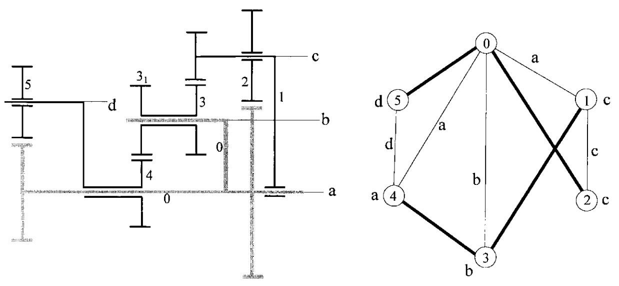


Fig. 10. Example of a gear system which is not a well-formed planetary gear system.

7. Conclusions

The incessant pressure to reduce time while increasing the quality of engineered products emphasises the importance of searching for every possible efficiency when making a mathematical analysis of an engineered system. There is benefit in analysing the well-formedness of the topology before progressing to solving the equations of the system. Topological analysis can usually be performed much more quickly than solving for the behaviour of the system, and if a topological impossibility is found before progressing to full analysis, much time and effort is saved. Topological consistency or well-formedness is analysed by using a mathematical representation which is isomorphic with the engineering system being dealt with. An isomorphic representation has the property that there is a one-to-one correspondence between elements of the mathematical representation and the physical system. Graph theory can be used as such an isomorphic representation, and matroid theory is then invoked as a means for proving various useful theorems on the representation.

Topological well-formedness is a necessary condition for the system to be physically viable and mathematically soluble, but it is not sufficient. If the definition of the system is not a well-formed topology, it will not be feasible. However, if it has a well-formed topology, singularities in the geometric or other state variables can still inhibit correct physical behaviour and mathematical solubility (as, for instance, in a mechanism at top-dead-center).

The paper showed how this representation is used to check the topological well-formedness of three types of engineering system: a structural truss, a planetary gear wheel system, and a dynamic mass-spring-damper oscillator. The method can be applied to many more types of engineering systems. Having checked the topological well-formedness, the full analysis can proceed with the assurance that if a singularity or similar impossibility occurs in the calculation, it will necessarily be because of the values of the geometric or other state variables in the system, and cannot be due to the topological configuration. This approach saves time and effort when computing the behaviour of engineering systems, especially if they are large and complex, with many elements.

Having completed the topological analysis of the system, the behaviour may be computed using traditional methods. However, it is often useful to compute the behaviour from the isomorphic representation of the system. This subject is not covered in the paper. The reader is referred to the papers in the literature dealing with this [5,6]. It is expected that more papers on analysis using graph theory and related representations will be published in the near future.

References

1. Goldman, S.L., Preiss, K. (Editors), Nagel, R.N., Dove, R. (Principal Investigators) (1992) 21st Century Manufacturing Strategy. Lacocca Institute, Lehigh University, Bethlehem, PA 18015. (Prepared for the US Congress)
2. Next Generation Manufacturing (1997) A report prepared for the U.S. National Science Foundation, obtainable from Leaders for Manufacturing program at MIT, the Agility Forum at Lehigh University, or Technologies Enabling Agile Manufacturing of the US Department of Energy
3. Preiss, K., Goldman, S.L., Nagel, R.N. (1996) Cooperate to Compete: Building Agile Business Relationships, van Nostrand Reinhold, NY
4. Shai, O. (1997) Representation of Embedded Engineering Knowledge for Artificial Intelligence Systems. PhD thesis, Ben Gurion University of the Negev
5. Fenves, S.J. (1966) Structural analysis by networks, matrices and computers. J. Structural Division, ASCE, 92, 199–221
6. Preiss, K., Shai, O. (1994) Deep artificial intelligence knowledge for truss analysis. The 25th Israel Conference on Mechanical Engineering, Haifa, Israel, 207–209
7. Kron, G. (1962) Elastic structures from the point of view of topological network theory, RAAG Memoirs, 3, 329–337
8. Kron, G. (1963) Diakoptics – the Piecewise Solution of Large Scale Systems, London, Macdonald
9. Fenves, S.J., Branin, F.H. (1963) Network topological formulation of structural analysis. J. Structural Division, ASCE, 89, ST4, 483–514
10. Lind, N.C. (1962) Analysis of structures by system theory. J. Structural Division, ASCE, 88, ST2, 1–22
11. Kaveh, A. (1992) Structural Mechanics: Graph and Matrix Methods, Chichester, John Wiley
12. Andrews, G.C. (1971) The Vector-Network Model – a Topological Approach to Mechanics. PhD thesis, University of Waterloo
13. Maxwell, J.C. (1864) On the calculation of the equilibrium and stiffness of frames. Philos. ag. Ser. (4), 27, 294–299
14. Laman, G. (1970) On graphs and rigidity of plane skeletal structures. J. Engineering Mathematics, 4, 331–340
15. Lovasz, L., Yemini, Y. (1982) On generic rigidity in the plane. SIAM J. Algebraic and Discrete Methods, 3, 91–98
16. Swamy, M.N., Thulasiraman, K. (1981) Graphs: Networks and Algorithms, New York, John Wiley
17. Tutte, W.T. (1961) On the problem of decomposing a graph into n-connected factors. J. London Mathematics Society, 36, 221–230
18. Shai, O., Preiss, K. (1998) Representation of embed-

ded engineering knowledge for design. In: Gero, J.S. Sudweeks, F. (Editors), 5th International Conference on Artificial Intelligence in Design, Kluwer Academic, 149–167

19. Erdman, A.G. (1993) Modern Kinematics – Developments in the Last Forty Years, New York, John Wiley
20. Freudenstein, F. (1971) An application of boolean algebra to the motion of epicyclic drives. ASME J. Engineering for Industry, 93, 525–532
21. Polomodov, B., Gershon, T. (1995) Matriculation Project: checking the validity and analysis of planetary gear system. Ort High School, Rehovot, Israel
22. Preiss, K., Shai, O. (1996) Line Graph Representation for Engineering Problems. 26th Israel Conference on Mechanical Engineering, Haifa, Israel, 604–607