

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

**A Transformational Approach to
Mechanical Design Synthesis**

by

J. R. Rinderle, S. Finger

EDRC 24-21-90 C. *J*

A Transformational Approach to Mechanical Design Synthesis

James R. Rinderle

Susan Finger

Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

During the design process, a designer transforms an abstract functional description for a device into a physical description that satisfies the functional requirements. In this sense, design is a transformation from the functional domain to the physical domain; however, this transformation process is not well characterized nor understood for mechanical systems. The difficulty arises, at least in part, because mechanical designs are often composed of highly-integrated, tightly-coupled components where the interactions among the components are essential to the behavior and economic execution of the design. Each component may contribute to several required behavior, and a single required system behavior may involve many components. In fact, most mechanical components perform not only the desired behavior, but also many additional, unintended behaviors. In good mechanical designs, these additional behaviors often are exploited.

The long term goal of our research is to create a transformational strategy in which the design specifications for a mechanical system can be transformed into a description of a collection of mechanical components. To realize this goal requires formal representations for the behavioral and the physical specifications of mechanical systems as well as formal representations for the behaviors and the physical characteristics of mechanical components. Because the interactions of components are important in our synthesis strategy, the representation of the behaviors of mechanical components must be linked to the representation of their physical characteristics; that is, we are concerned with modeling the relationship between form and function of components. Finally, we need a strategy that enables us to transform an abstract description of the desired behavior of a device into a description that corresponds to a collection of available physical components.

We present a graph-based language to describe both the behavioral specifications of a design as well as the behavior of the available physical components. We also demonstrate how the specification graph may be transformed so as to correspond to collections of available components.

Introduction

During the design process, a designer transforms an abstract functional description for a device into a physical description that satisfies the functional requirements.¹ In this sense, design is a transformation from the functional domain to the physical domain [Mostow 85, Rinderle 82]; however, the basis for selecting appropriate transformations and methods for accomplishing transformations are not well understood. The implicit basis for design transformations in circuits [Steinberg 86], software [Winh 71], and some architectural applications [Fenves 87] result in a degree and type of modularity not well suited to mechanical devices [Rinderle 86].

¹Mechanical engineer* tend ID aw Ae words nñctáo and behavior interchangeable. Qualitative physicists make a distinction between these words; that is, the design's *function** is what it is used for, while its *form* is what it does. *Function* might be the location of hands. Similarly, a *form* may be designed to function as a prime mover, but can also *function* as a door flop because it has additional behaviors due to its mass. In this paper, *form & function* is used to indicate the subset of behaviors which are required for the device to perform automatically.

Consider the design of a simple gear box as an example of a transformational approach to design and one specific approach to selecting and applying transformations. The gear box must provide a 25:1 speed reduction, a right angle configuration between the input and output shafts, and an offset between the input and output shafts. These three functional requirements are shown in Figure 1. The speed reduction requirement is further decomposed into a pair of speed reducers as shown in the figure. Each of the lowest level requirements is then transformed into a collection of physical devices. Each of the 5:1 speed reductions are transformed into spur gear pairs. The right angle requirement is transformed into a miter gear set and the offset requirement is transformed into a 1:1 spur gear set. These four sets of gears may be combined to achieve the overall device functionality. The device as a whole consists of three spur gear sets, and one miter gear set, totaling 8 separate gears. This solution is obviously large, cumbersome, and costly, particularly compared to a single worm gear set, such as that shown in Figure 2. The worm gear set accomplishes the steep speed reduction, the right angle requirement and the offset shaft requirements all in a single device.

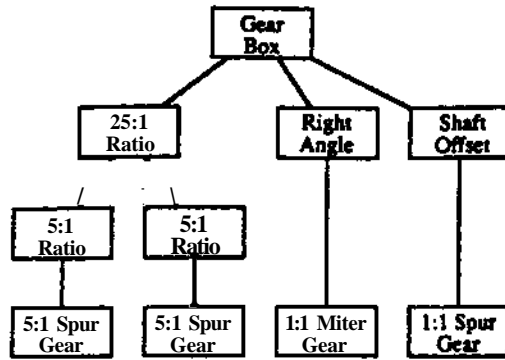


Figure 1: Functional decomposition and transformation of a gear box specification

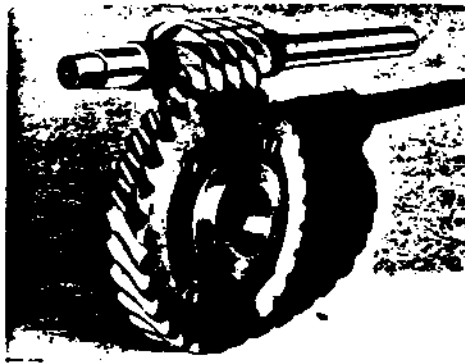


Figure 2: A worm and worm gear combination which meets all specifications given in Figure 1. (Courtesy of Horsburgh and Scott Company)

While the method just illustrated provides a transformational approach, it is not sufficient to achieve satisfactory mechanical designs.

Good mechanical designs are often composed of highly-integrated, tightly-coupled components where the interactions among the components are essential to the behavior and economic execution of the design. This assertion runs counter to design methodologies in other engineering fields, such as software design and circuit design, that result in designs in which each component fulfills a single function with minimal interaction. Because of the geometry, weight, and cost of mechanical components, converting a single behavioral requirement into a single component is often both impractical and infeasible. Each component may contribute to several required behaviors, and a single required system behavior may involve many components. In fact, most mechanical components perform not only the desired behavior, but also many additional, unintended behaviors. In good mechanical designs, these additional behaviors often are exploited.²

The direct transformation of behavioral requirements into physical components may result in undesirable designs for two reasons. The first is that matching individual behaviors directly to components

²This statement does not contradict the design axioms put forth by Suh [Suh 80, Suh 88]. The design axioms state that good designs maintain independence of functional requirements and minimize the information content of the design. Suh points out that by integrating functions into a single component, information content may be reduced without compromising the independence of functional requirements.

does not enable the integration of behaviors into more economical or more economical descriptions of physical components. The second reason is that the behaviors of components have not only the desired behavior and physical characteristics but also many additional incidental behaviors and characteristics. The appropriate device configuration and selection of components depends to some extent on the incidental behaviors.

By creating a formal description of a limited set of behaviors for mechanical designs and a corresponding description of physical components, we can generate the description of a physical system that takes advantage of the multiple behaviors of its components. This paper focuses on the grammar that underlies the transformation from behavioral specifications to physical components. The transformation strategy is discussed in [Hoover 89].

Overview of Our Approach

The goal of our research is to create a transformational strategy by which the design specifications for a mechanical system can be transformed into a description of a collection of mechanical components. Both behavioral and physical requirements as well as behavioral and physical characteristics of the available mechanical components must be represented to execute such a transformational approach to design. We are investigating the use of representations based on formal grammars to facilitate the characterization of our approach with respect to completeness, complexity, etc. and to take advantage of the advances in formal language theory. Because the interactions of components are important in our synthesis strategy, the representation of the behaviors of mechanical components must be linked to the representation of their physical characteristics; that is, we are concerned with modeling the relationship between form and function of components. Finally, we need a strategy that enables us to transform an abstract description of the desired behavior of a device into a description that corresponds to a collection of physical components. This paper primarily addresses the first issue, that of formal representations of behavior.

To realize the goal of formalizing the transformation from the behavioral to the physical domain, we have begun to explore a small domain within mechanical design, the domain of gear box design. Clearly, one reason for selecting this domain is that gear box design is a well-understood, highly-parameterized area of mechanical design. Nevertheless, we believe that our representation and transformation formalism will be applicable in other mechanical design domains, particularly to the class of design problems that we call *configuration design*. By configuration design, we mean designs composed from standard component families but for which allowable configurations are not specified a priori.

Our approach is based on the following assertions:

- The behavioral requirements of mechanical systems can be represented using a graph grammar based on bond graphs.
- The behavioral characteristics of components can be represented using a graph grammar based on bond graphs.
- The physical characteristics of designs and components can be represented using an augmented topology and geometry graph.
- The behavioral and physical graphs of components can be linked parametrically.
- The behavioral specifications graph can be transformed into a description of a physical system with associated behavioral and geometric representations.

Representational issues will be discussed at greater length in subsequent sections; however, we note here that our underlying representation for behavior is based on bond graphs [Petya 61]. Using bond graphs, we can construct a formal grammar that gives us a general representation of classes of mechanical behavior. In common practice, bond graphs are constructed to model the behavior of physical systems. We use bond graphs not only to model the behavior of physical systems, but also to represent behavior in the abstract, as with a design specification. Thus a device configuration can be generated by transforming a specification bond graph into a functionally equivalent graph which corresponds to a configuration of available components. The type of graph transformations used are those that decompose, aggregate, and redistribute graph primitives.

A major advantage of using bond graphs to represent design

lequeinimi is that we can define unantfirmarion roles that alter the structure of the bond graph but that do not alter the **dynamic behavior of the system represented by the graph. The implications of this statement are important. Because we can transform the** tpecrfVarifms graph to represent many different physical systems, we do not inmpoae an initial structure or configuration on the physical design; that is, we do not require an a p r i ^ decomposition of ite design specifications.

To complete this methodology, we plan to represent the physical characteristics of designs and components using another graph grammar that is based on an aligned topology mph [Pinilla 89J and non-manifold geometry [Gursox 89). We plan to link parametrically the bond graph representation and the topology and geometry graph representation, with a geometric i^iw^jcnomoi, characteristics such as the volume or mass can be modeled and computed, and from the bond graphs, the dynamic behavior of the final design can be modeled. u this paper, we do not discuss the geometric representation in detail because our work on linking the geometry and behavior is preliminary.

A Brief Introduction to Bond Graphs

Bond graphs, which were created by Paynter [Paynter 61], provide a convenient and uniform representation for the dynamic behavior of a broad class of physical systems, including those within the mechanical, electrical, hydraulic, thermal, and biological domains. Bond graphs have been used extensively in a wide variety of application areas including robotic manipulators [Margolis 79], torque converters [Hrovat 85], and vacuum cleaners [Remmerswaal 85]. A brief introduction to bond graphs is given here. For a complete discussion, see Karnopp and Rosenberg [Karnopp 75].

Bond graphs enable mechanical and hydraulic systems to be represented in a manner equivalent to electric circuit diagrams. For example, a spring in a mechanical device acts like an electrical capacitor by storing and releasing energy. To illustrate the similarity, Figure 3 shows an RLC circuit with its equivalent bond graph and a mass-spring-damper system with its equivalent bond graph. While Figure 3 illustrates separate electrical and mechanical systems, one of the most powerful attributes of bond graphs is that they can be used to model integrated electrical, mechanical, and hydraulic systems.

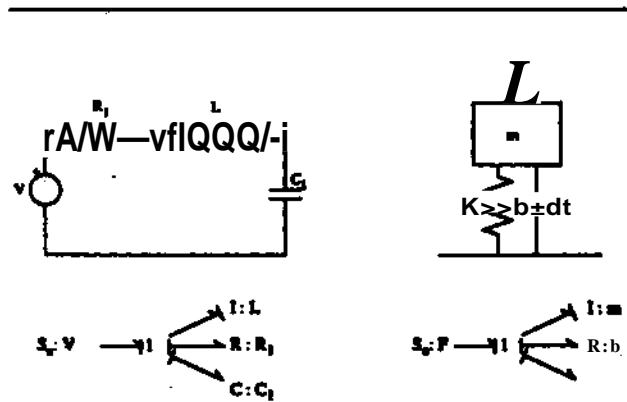


Figure 3: Simple systems with corresponding bond graphs

Using bond graphs, physical systems are represented as a graph of lumped-parameter, idealized elements. Power is the currency of bond graphs; power flows through the bonds (edges) in the graph, and power is dissipated, stored, supplied, and transformed at die ports (vertices) in the graph.

The ports, or vertices, of bond graphs are divided into three categories:

- 1-port elements dissipate power, store energy, and supply power. Dampen, springs, and masses are die mechanical elements represented by the passive 1-port elements. Force (effort) and velocity (flow) sources are represented as active 1-ports.
- 2-port elements transform power. Transformers are 2-

port elements that represent an imposed proportionate relationship between similar quantities, e * a fear pair constrains rotational speeds. Gytrators are a 2-port elements that impose t proportionate relationship between dual quantities, eg. a torque converter constrains the relationship between torque and angular velocity. Ifcwer is conserved across a 2-port

- N-port dements represent the structure of the system corresponding to the connections among the elements. There ate two types of N-port elements: exjunctions and 1-junctions, which correspond respectively to "same force" and "same velocity" connections. Equivalents of KirkoJT's laws apply to N-port elements: the sum of the efforts around a 1-junction is zero (the bonds share a common flow); die sum of the flows around a 0-junction is zero (the bonds share a common effort). N-port elements are power conserving.

Because bond graphs can be used to model electrical, mechanical, and hydraulic systems, problems with standard notation and terminology arise. For example the symbol, v, is commonly used for *velocity* in mechanical systems and for *voltage* in electrical systems. In circuits, it U common to speak of a voltage source, but speaking of a force source in mechanics is awkward. Table 1 gives a common notation for bond graph equivalents in the electrical and mechanical domains.

	Generalistd Bead Graphs	Electric* Sjtum*	Mechanical v, p, m
Variables	effort.*	Vokif**	Force, P
	ffew.f	Qmmi	V<Jod*.w
	Power = ef	IW.vi	Pmv.Fv
Source Elements	Effort Sow*	VokiftSare	AppMFm
	Flow Sara	Oral Sore	Praoibad Velocity
Pm>ve Element!	Resistor E •<ftf	Resistor R v-ift	Dmp* i F-bv
	Capacitor C r-ctfv/ui	Of*** C- {•Cdv*	dFM<kv
	Inertance I ••Ida*	Inductor L vLdtf*	bmxm m F><<*tti
StoKtavr	0 Junction OHHTHHT FFFwi	P n U	Imm Km*
	1 Junction GMSMBROW	Smim	S<MV4ac*y

Tablet: Equivalents for electrical, mechanical ind generalized bond graph elements

The bond graphs shown in Figure 3 consist of 1-port elements, n-port elements, and power bonds. Power bonds are indicated by the presence of a half-arrow, the direction of which establishes the sign convention for power flow. Power bonds are usually shown with a causal stroke at one end. The causal strokes play an important role in identifying device states, formulating system equaoons, and interpreting causal structure, but they, like the power direction half-arrow, may be considered as augmentations to the bask bond graph. Although they are included in the illustrations, neither the causal strokes nor the half-arrows enter into representational or grammatical considerations. Other bond graphs elements, notably field structures and active bonds, are not considered here.

TTie use of bond graphs in our representation and grammar will be discussed in following sections.

Related Work

Our research builds upon research from several different areas including representation of mechanical behavior, generative design, and configuration design. In this section, we briefly discuss related work in these areas.

Representation of Function

A transformational strategy for design requires a representation of the device specifications and the components used in the device. The representation of the function and behavior of mechanical designs has been explored by, among others, Lai [Lai 87], Oossfey [Oossfey 80], Pahl and Beitz [Pahl 84]. The function structures of Pahl and Beitz provide a graphical system for laying out the functions of a design. In this system, functions such as "mix" or "deliver" are arranged in a graph to represent the overall function of the design. This is similar to our idea of a specifications graph, discussed below; however, there are several important differences. Pahl's work does not discuss how to integrate form specifications or functional constraints with this functional representation. Therefore, while the function structures are used to study various configurations in Pahl's synthesis strategy, there is little guidance for transforming the function structure to a physical description of the device. Lai has created a formal, English language-based system called FDL for representing the function and structure of mechanical designs. In FDL, nouns and verbs are used to create sentences that represent the function of a design, and design rules operate directly on the nouns and verbs in the sentence. Allowable verbs, for example "fasten," do not have physical or mathematical representations and so their meaning is determined by the rules that use them. While the FDL language can represent the function and form of a design it provides no assistance in transforming a functional description into a physical description.

More closely related to our approach for representing behavior is the work of Fenves and Baker (Fenves 87) and of Ulrich and Seering [Ulrich 87, Ulrich 88, Ulrich 89]. Fenves has created a spatial and functional representation language for structural designs. They use operators that execute a known grammar to generate architectural layouts as well as structural and functional configurations; however, they assume that the layout and structure are independent if they are generated sequentially. Ulrich and Seering also use a formal representation of function based on bond graphs; however, their goal is to create a system in which the bond graphs for single-input, single-output dynamic systems can be automatically synthesized and transformed into physical components. Using a strategy they call *design and debug*, they transform a graph of design requirements directly to functionally independent physical components. Reconfiguration for function sharing is performed after the components have been selected.

Grammars for Shape Representation

Formal grammars that can be used to represent, generate, and parse valid strings in a language have proven useful in a number of fields, most notably, linguistics and computer science. Recently, interest has been growing on the use of formal grammars in engineering design. Our work draws mainly from these engineering applications of grammars. We are specifically concerned with graph grammars, the class of grammars that operates on graphs. Tutorials on graph grammars and their applications are given by Ehrig [Ehrig 87] and by Nagl [Nagl 87].

One of the earliest uses of grammars in design was by Stiny [Stiny 75] who created shape grammars based on the formalisms of computational linguistics (Chomsky 57). Architects in particular have been interested in shape grammars, using them to generate a family of floor plans or ornamentation. Fitzhorn [Fitzhorn 89] has shown the formal relationship between language theory and solid modeling systems. He shows that a variant of a graph grammar can produce three-dimensional solids. He creates three grammars, one of which generates the constructive solid geometry representation, the second of which generates the boundary representation, and the third of which generates plane models.

Pinilla *et al.* [Pinilla 89] have created a grammar that can be used to describe and represent the geometric features of a design. They use a non-manifold topological representation of a design to create a general, but formal, representation of form features. This system is currently being extended to enable feature-based designs to be generated, represented, and parsed. This extension is possible because the underlying representation of a feature is based on elements of a well-defined grammar.

Configuration Design

Configuration design and parametric design are active areas of research in mechanical engineering design. For a more complete work, see [Finger 89]. Our research combines configuration design and parametric design because we generate both the structure of the design and the individual components. Many design systems, such as HI-RISE [Maher 85], AIR-CYL [Brown 85], and VT [Marcus 86], utilize either a set of predetermined decompositions of the structure of a design, or utilize design methods that generate, as part of the design process, a decomposition belonging to such a set. Therefore, all of the designs generated by these systems will share, at a relatively low level of structural similarity. For design domains in which the most desirable structures can be enumerated or explicitly decomposed in advance, this approach proves useful. But there are many design problems where the structural decomposition of the most useful solution is not predetermined.

Representation of Specifications, Components, and Devices

The transformational approach to design, which we have briefly introduced, imposes some representational requirements. Specifically, the representation must

- Express formally the design requirements of mechanical systems.
- Be compatible with the representation of the behavior of components.
- Facilitate design validation.
- Represent the required system behavior without imposing a pre-defined structure on the physical realization of the design.

We make the observation that system specifications for engineering designs are of two types; *behavioral* and *physical*. That is, some specifications describe at an abstract level the desired behavior of the overall system while others describe physical restrictions or requirements on the final design. For example, the requirements for a vibration absorber might include the *behavior* of the device in terms of frequency and rejection ratio and might also specify *physical* properties such as allowable size and weight. The physical and behavioral specifications express the design objective. Since the specifications are given without regard to design configuration, they are independent of each other.³ Physical specifications for a design may or may not be given, while at least some behavioral specifications must be given, since the behavioral specifications express the central aspect of the design objective.

Although the behavioral and physical specifications are independent in the functional domain, they are coupled in the physical domain, because any physical arrangement results in a specific set of behaviors. In the physical domain, the physical and behavioral characteristics of an individual component depend on one another, and the behavior of the whole design depends strongly on the configuration and interaction of components. The representation of interactions is essential for our purposes since we are investigating the effects of and the means for achieving functional integration in design.

Physical and behavioral specifications and characteristics can be represented as combinations of abstract primitives. They are abstract because each primitive corresponds to only one behavioral or physical characteristic. Individually they do not correspond to any particular component or configuration of components, but collectively they may represent the design specifications or the form and behavior of components. There are two important criteria that a set of primitives must satisfy. One requirement is that the set of primitives chosen be complete; that is, all relevant behavioral and physical characteristics must be representable by some combination of primitives. The other is that the number of primitives must be small, although not necessarily minimal. The latter requirement minimizes combinatorial problems associated with representing a single behavior in different ways. In addition, the primitive behavioral and physical elements should enable commonly used components and typical specifications to be represented easily.

³That is, they are independent unless the requirements are contradictory, for example by virtue of physical laws.

Representation of the Design Specification

If the behavioral primitives correspond to bond graph elements, the behavioral specifications can be represented by a bond graph. This specification bond graph is not unique, indeed any behaviorally equivalent graph is an acceptable representation of the specifications. The specifications graph represents only the desired behavior of the design. Physical specification will be represented as an augmented topology graph which is not discussed further in this paper.

Deriving a correct specifications graph for a design problem is, in itself, a major research problem. Initially, for transmission design, we assume that deriving a specifications graph from the problem statement is straightforward. Later, as less constrained design domains are explored, we will look at the problem of constructing the specification graph from the design requirements.

Figure 4 illustrates the system specification for a system required to drive two loads at a high speed relative to the input speed. The single physical requirement is a required shaft offset and is not shown.

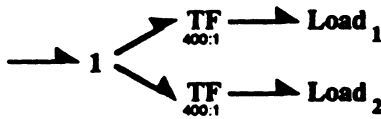


Figure 4: Specification graph for a system to drive two loads

Representation of Physical Components

Physical components, like specifications, have both behaviors and physical characteristics. Therefore, it is convenient in representing components to employ the same basic structure that is used for specifications. One crucial difference, however is that the behaviors and physical characteristics of a component are inherently linked; no single characteristic can be obtained in isolation. Each physical component is represented as an object that has a behavioral representation, a physical representation, and an explicitly represented interaction between the two. The relationships can take the form of design equations, analytical models that relate geometric characteristics to behavioral characteristics, or data base entries that prescribe a relation between the physical and the behavior characteristics. For example, the weight of a helical coil spring, which is a physical characteristic, is proportional to the product of stiffness and the square of allowable deflection, which are behavioral characteristics. These relationships are often crucial during the design.

As an example consider a single spur gear as a component. The behavior graph for the component is shown in Figure 5. In this example, a relationship is imposed between flow quantities rotational speed, ω on the left and surface speed, v , on the right. The topological graph, representing physical characteristics is not shown. In its place we simply show the geometric parameter corresponding to pitch diameter.

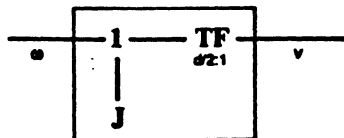


Figure 5: Behavior graph for a single spur gear

It is important to note that the completeness of the behavioral and physical representations determines the extent to which the additional behaviors can be exploited. For example, with a worm

gear, the right angle shaft configuration and the self-locking property must be represented explicitly, or be derivable from the representation, for it to be recognized and utilized in the design process. Therefore, this methodology requires a known component base.

Representation of Designs

A complete design is the specified configuration of a set of components. Since the components have associated with them both behaviors and physical characteristics, the distinction between the representation of designs and the representation of components is in the representation of the behavioral and physical configuration. The behavioral configuration fundamentally consists of the kinematic connections, e.g. mounting to a frame, rigid connections, or rolling. Most of the common kinematic arrangements can be categorized [Reuleaux 54] and reduced to a bond graph junction structure. A 1-junction, for example, represents a common translational velocity. In this way, the behavior of complete devices may be represented in terms of the behavioral bond graphs of components and a number of bond graph junction structures representing kinematic connections. Partially complete designs also consist of components and junction structures and some number of behavioral primitives not yet associated with a component.

Ultimately, we will represent the physical characteristics of the device in much the same way, i.e. as a composite of the topology and geometry graphs of the components and configuration. To date we have represented the physical characteristics of devices by aggregating, on an *ad hoc* basis, the geometric parameters of the components and the layout. Figure 6 shows how two spur gears are represented as a combination of components joined through a kinematic connection. In this case, the components have a common speed at the pitch diameters of the gears.

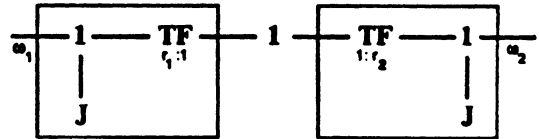


Figure 6: A behavior graph for a combination of two spur gears

At the device level, a physical or behavioral characteristic describes an overall characteristic of the system. These characteristics may be one of two types: configuration-independent and configuration-dependent. Configuration-independent characteristics are those such as weight that depend only on the selected components and not on their configuration or interactions. Configuration-dependent characteristics, such as size and resonant frequency, depend on the configuration and interaction of the components in the final design. In either case the characteristics of the design can be determined from the completed behavioral and topological graphs, however, the implications of configuration independent characteristics can be utilized prior to design completion.

Bond Graph Transformations

In this section, we discuss the formalisms that enable bond graphs to be transformed from design specifications to component specifications while preserving the system behavior. We first define the graph, its elements, and some set operations on the graph such as inclusion and difference. We then define what it means for one graph (the component graph) to be derivable from another graph (the specification graph). In this paper, our primary interest is in the productions for transforming bond graphs of mechanical systems, rather than in demonstrating a grammar that generates all possible, valid bond graphs.

Graph Representation

Behavioral graphs, U , bond graphs without causal and power sign augmentation⁴, are undirected graphs. Each node of the graph is labeled to indicate its type, e.g. inertial, energy storage or common velocity junction. Each of the edges of the graph is a power bond connecting two vertices. The constitutive laws for each of the elements impose requirements under the bond graph structure. Principle among these is the connectivity of the 1-port, 2-port and n-port elements. An additional requirement is the consistent imposition of source elements. These requirements are easily captured in a bond graph generation grammar. Such a grammar may also be used to parse a bond graph representing a behavioral specification to insure that a specification is valid. After the specification graph has been validated, the graph can be incrementally transformed into a behaviorally equivalent graph which more closely corresponds to collections of the available physical components.

A bond graph is a graph with vertices labeled to indicate correspondence to the various 1-ports, 2-ports and n-ports. The edges of the graph correspond to the power bonds and may be labeled to show, for example, causality. Loops may occur in a bond graph but multiple edges and self edges are not meaningful and are not allowed.

Basic Graph Relationships and Operations

Transformations of the specifications graph will require removing subgraphs of bond graphs and replacing them with other graphs. For example, if a specification included a graph representing a transformation ratio of 25:1, that subgraph would be removed, and a graph representing a sequence of 5:1 reductions might be inserted in its place. To perform such a transformation requires a definition of graph isomorphism so that different arrangements of the same graph can be recognized. It also requires a definition of graph inclusion so that subgraphs within a larger graph can be identified, that is, for example, to define what it means to say that the graph of the 25:1 reduction ratio is included in the specifications graph. Finally, it requires a definition of the difference operation between two graphs so that we can, for example, remove the 100:1 reduction from die specifications graph in order to replace it with the sequence of smaller reductions. Graph isomorphism, inclusion and difference are defined in Appendix A and are illustrated in Figure 7.

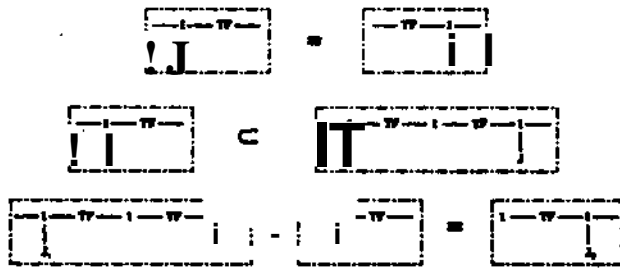


Figure 7: Graph isomorphism, inclusion, and difference

Graph Productions

We may define a grammar to operate on any valid specification bond graph. Specification bond graphs can be thought of as the start symbol of the grammar and the productions in the grammar are all transformations on bond graphs that preserve the function of the graph.⁴ The set of valid productions is without bound; however, many productions can be selected which facilitate the decomposition of high-level functional characteristics into collections of lower-level characteristics or which recombine, aggregate, or redistribute bond graph primitive elements. These transformations correspond to a rearrangement, aggregation, or decomposition of the physical

⁴The distinction between preserving *function* and preserving *behavior* is significant. We do not address this issue here.

components in a device. Therefore, the selection of an appropriate graph production is critical.

We now define graph productions that enable us to remove one part of the bond graph and replace it with another. A graph production consists of a right hand side, a left hand side and an embedding transformation. That is, a production, P , transforms the left-hand graph, f , into the right-hand graph, g .

$$P = (g, f, E)$$

where E is the embedding transformation that gives the procedure for inserting the right-hand graph into the parent graph after the left hand side has been removed. In the productions defined so far, the embedding transformation involves a simple reconnection of bonds in the right-hand graph to vertices in the parent graph, so the embedding transformation will not be defined formally in this paper. A graph production is illustrated in Figure 8.

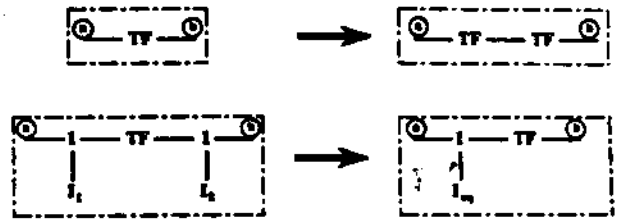


Figure 8: Graph productions

Example of a Transformation from Specifications to Components

In this section, we illustrate how the graph productions are applied to a specifications graph resulting in a design alternative. In this paper, we do not discuss the strategy for selecting the sequence of transformations. The strategies are a topic of current research and are discussed in [Hoover 89]. Briefly, the transformation strategy selects and configures components into a physical system based on two characteristics typical of good mechanical designs:

- Functional integration
- Utilization of incidental behaviors

Transformations that result in increased functional integration create a device configuration in which a single component contributes to more than one of the behavioral requirements of the design. Transformations that result in increased utilization of incidental behaviors create a device configuration in which secondary behaviors or physical characteristics of components are exploited.

Behavior-Preserving Transformations

The specification graph is not unique since there are many graphs which represent the same behavior. Figure 9 shows a specification graph for the design of a gear box that must drive two loads at a speed 400 times as great as the input speed. Other specification graphs are valid and, as many will immediately observe, the particular specification graph given is not likely to be the most convenient.

By transforming the specification graph, without altering behavior, we can explore design alternatives that have the same behavior. Some of the resulting configurations result in physically desirable designs and some do not. In addition to knowing the general rules for behavior-preserving transformations, we need to know which transformations to apply and the sequence of application. Guidance in selecting the behavior-preserving transformation typically comes from the physical requirements of the system, from the physical characteristics of the components, and from the relationship between geometry and behavior of the component.

The transformation process described in [Hoover 89] is guided by the function integration and incidental behavior principles and by knowledge of the available components. It provides an approach to finding configurations that meet the physical requirements, thus

eliminating blind search. This approach has been successful for the design of mechanical power transmissions and serves as a means for extending this technique to other domains.

The specification graph is composed of abstract behavioral primitives that do not yet correspond to any physical components. To arrive at a design we transform the specification graph* without changing function, so as to obtain a graph that more nearly corresponds to a collection of components. Figure 10 shows the resulting graph in which each of the rotational-rotational transformers has been replaced with a pair of rotational-rotational transformers that might correspond to individual spur gears.

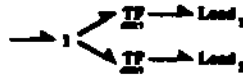


Figure 9: Specification graph for two-load design requirement

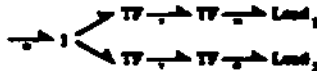


Figure 10: Transformed specification graph of Figure 9

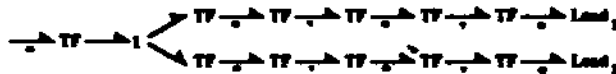


Figure 11: Component-directed splitting transformation of Fig. 10

Component-Directed Transformations

Because mechanical designs are characterized by a high degree of integration, transformations should be directed toward this goal. Integration in a design requires the appropriate utilization of the behavioral and physical characteristics of its components. Therefore, intelligent application of transformations requires utilizing both the behavior and geometric graphs of the components to guide the selection process. Transformations selected in this way are *component-directed transformations* because their selection and use are directed by knowledge of the available components. It is the class of component-directed transformations that ultimately enables the selection of a single component to fulfill multiple functions, e.g. selecting a worm gear to execute speed reduction, shaft offset and right angle functions as described more fully in [Hoover 89].

In this paper we have restricted the component domain to spur gears; nevertheless, the component directed transforms are important. The pair of transforms shown in Figure 10, if mapped directly to components would require a pair of gears with a 400:1 diameter ratio, which is clearly not practical. Directed by the allowable size range of available components, we apply a splitting transform to arrive at the graph shown in Figure 11.

The individual transformers in Figure 11 can then be mapped to spur gears (see Figure 5). The resulting design is shown in Figure 12. Note that as the mapping transforms are executed the dynamic behavior of the device changes due to the introduction of gear inertia. Because the *functional* the transmission is kinematic rather than dynamic, these mapping transforms are function preserving.

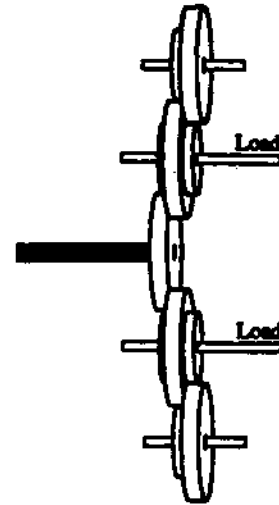


Figure 12: Physical system corresponding to figures 9, 10, and 11

Other Transformations

Although the gearing system shown in Figure 12 is feasible, it is overly large* costly* and complex because there is no commonality of the power paths. Component-directed transforms do not have the scope to identify this type of deficiency, however* the graph production shown in Figure 13 does increase power path commonality. It is one production in a class of transforms which extend the function integration concepts to the device as a whole. Again, without discussing the means of selecting transforms we point out that the graph shown in Figures 9, 10, or 11 can be transformed in a function preserving manner to the one shown in Figure 14.



Figure 13: Graph production used to achieve a common power path

This specification can be mapped to an arrangement of gears as shown in Figure 15. This configuration has five fewer gears, one less shaft and is significantly more compact than the configuration shown in Figure 12.

The identification and selective application of this type of transformation is one of the most important and challenging aspects of our strategy for mechanical design.

Results and Future Work

We have presented a behavioral and physical representation of the specifications and components for a limited class of mechanical designs. For transmission design* we have created the component database that contains graphs and form-behavior relations for bevel, spur, and worm gears, and we have implemented the transformation strategy discussed in [Hoover 89]. However, this paper presents preliminary work that must be expanded in many directions before we can prove that our transformational approach is valid for a larger class of mechanical designs.

Our research will continue in the following areas:

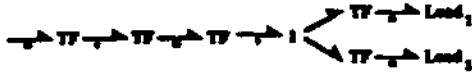


Figure 14: An alternative graph corresponding to Figure 11

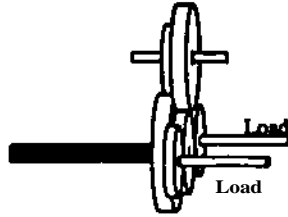


Figure 15: An alternative design corresponding to the specification given in Figure 9 and to the transformed graph shown in Figure 14

- Expanding the behavioral and physical characteristics that can be represented
- Continuing to develop the representation of the interconnection between geometry and behavior.
- Increasing the number of components from different mechanical design domains represented in our system, and representing them at higher-levels of abstraction as well as at greater levels of detail.
- Creating new transformation strategies, refining the existing ones, and in particular, creating general transformations that operate based on system-level characteristics rather than on component characteristics.

Acknowledgments

The authors are pleased to acknowledge the support of the Design Theory and Methodology Program of the National Science Foundation (NSF Grants DMC-84-51619 and DMC-88-14760) and the Engineering Design Research Center at Carnegie Mellon University (NSF Grant CDR-85-22616).

References

- (Brown 85)
 Brown, D. and Chaidrasekaran, B., "Expert Systems for a Class of Mechanical Design Activity." in *Knowledge Engineering in Computer-Aided Design*. Gcro, J., ed., North Holland. 1985. pp. 259-290.
- [Chomsky 57]
 Chomsky, R., *Syntactic Structure*. Humanities Press. Atlantic Highlands, NJ. 1957.
- [Crossky80]
 Crossky, E., *Technical Representation Language for Structural Design*. "Expert Systems in Computer-Aided Design, Elsevier Science (North-Holland). IFIP 521987.
- [Ehrig87]
 Ehrig, H., "Tutorial Introduction to the Algebraic Approach of Graph Grammars," *Graph-Grammars and their Applications to Computer Science*. Springer-Verlag, New York. Lecture Note Series 1987. pp. 3-14.
- [Fenves87]
 Fenves, S. J. and Baker, N. C., "Spatial and Functional Representation Language for Structural Design." *Expert Systems in Computer-Aided Design*, Elsevier Science (North-Holland). IFIP 521987.
- [Finger 89]
 Finger, S. and D'Uono, L., "A Review of Research in Mechanical Engineering Design." *Research in Engineering Design*. Vol. 1. No. 1. 1989.
- [Rtzhorn89]
 Fithora, P., "The Formal Language of Solid Representation." *Out for review in*

Design Computing. 1989.

(Gurso.89)

Gurso, R. L. and Prim, F. B., "Corner-Based Representation of Surface Boundaries in Geometric Modeling." *Technical Report AIAA-89-1141*. American Institute of Aeronautics and Astronautics, 1989.

(Hoovers?)

Hoover, S. P. and Rindert, J. R., "Synthesis Strategies for Mechanical Devices." *Submitted to Research in Engineering Design*, 1989.

[Hrovat85]

Hrovat, D. and Tobkr, W. E., "Bond Graph Modeling and Computer Simulation of Automotive Torque Converters." *Journal of The Franklin Institute*, 1985, pp. 534-544.

[Karaopp75]

Kamopp, D. and Rosenberg, R., *System Dynamics: Modeling, Analysis, and Control*. John Wiley & Sons, New York, 1975.

[Lai87]

Lai, K. and Wilson, W. R. D., "TDL: A Language for Fact Description and Rationalization in Mechanical Design." *Computers in Engineering*. ASME, New York, 1987. pp. 87-94.

[Maher85]

Maher, M. U., "HI-RISE and Beyond: Direction for Expert Systems in Design." *Computer-Aided Design*, Vol. 17, 1985. pp. 420-427.

(Marcus86)

Marcus, S., Stouffer, U. and McDermott, J., "Elevator Designer that Uses Knowledge-Based Backtracking." Technical Report OAC-86-169. Department of Computer Science, Carnegie Mellon University, 1986.

[Margolis79]

Margolis, D. L. and Kamopp, D. C., "Bond Graphs for Flexible Multibody Systems." *Transactions of the ASME*. Vol. 101. 1979. pp. 50-57.

[Moore85]

Moore, J., "Toward Better Models of the Design Process." *The AI Magazine*, Spring 1985.

[Nagl87]

Nagl, M., "Set Theoretic Approach to ID Graph-Grammars." *Graph-Grammars and their Applications to Computer Science*, Springer-Verlag, New York. Lecture Note Series 1987. pp. 41-54.

[Pahl84]

Pahl, G. and Beitz, W., *Engineering Design*. The Design Council. Springer-Verlag, London. 1984.

[Paynter61]

Paynter, H. M., *Analysis and Design of Engineering Systems*. MIT Press, Cambridge, MA, 1961.

[Pinilla89]

Pinilla, J. M., Finger, S. and Prim, F. B., "Shape Feature Description and Recognition Using an Augmented Topology Graph Grammar." *1939 NSF Engineering Design Research Conference*. University of Massachusetts, Amherst MA. June 1-4, 1989.

(Remmerswaal 85)

Remmerswaal, J. A. M. and Paccjka, H. B., "A Bond Graph Computer Model to Simulate Vacuum Cleaner Dynamics for Design Purposes." *Technical Report AIAA-85-1141*. Franklin Institute. 1985. pp. 83-91

[Reuleaux 54]

Kennedy, A. B. W., editor. *The Kinematics of Machinery*. Dover Publications, New York, 1854.

[Rinderle 86]

Rinderle, I. R., *Measures of Functional Coupling in Design*, PhD dissertation. Massachusetts Institute of Technology, June 1986.

[Rinderle 86]

Rinderle, I. R., "Implications of Functional Coupling on Design Decomposition Strategies." *Computers in Engineering*, 1986, American Society of Mechanical Engineers, Chicago, 1986, pp. 193-198.

[Steinberg 86]

Steinberg, U., Langrana, R., Mitchell, T., Mosiow, J. and Tong, C., "A Domain Independent Model of Knowledge-Based Design." Technical report AWSL1 Project Working Paper No. 33. Rutgers University. March 1986.

[Stiny75]

Stiny, G., *Pictorial and Formal Aspects of Shape and Shape Grammars*. Birkhauser, Basel. 1975.

Safc.RP., Wilson. D.JL, Tka. W. W., Yasi*«, M. and BdI.
A. C. "Application of Axiomatic Design Techniques to Manufacturing."
Production Engineering Division, Winter Annual Meeting, American Society of
Mechanical Engineers, Washington, D.C., 19J01

[Suh88]

University Press, Oxford, UK, 1988.

[Ulrich 87]

Ulrich, K. and Seering, W. P. *Design: Synthesis of Systems
Components, Intelligence, and Integrated Manufacturing Systems and
Synthesis*. American Society of Mechanical Engineers, New York, 1987, pp.
57-66.

[Ulrich88]

Ulrich, K. T. and Seering, W. P., "Function Synthesis in Mechanical Design,"
National Conference on Artificial Intelligence, AAAI48, Minneapolis,
MN, August 21-26 1988.

[Ulrich89]

Ulrich, K. T. and Seering, W. P., "Synthesis of Schematic Description in
Mechanical Design," *Research in Engineering Design*, Vol. 1, No. 1, 1989.

[Winth71J]

Winth, N., "Program Development by Stepwise Refinement," *Communications of
the ACM*, Vol. 14, No. 4, 1971, pp. 221-227.