

# Bond Graph Based Automated Modeling for Computer-Aided Design of Dynamic Systems

**Zhaohong Wu**

Technology Development Department,  
Marine Acquisition Product Line,  
CGGVeritas Inc.,  
Houston, TX 77072  
e-mail: zhaohong.wu@cggveritas.com

**Matthew I. Campbell**

e-mail: mc1@mail.utexas.edu

**Benito R. Fernández**

e-mail: benito@mail.utexas.edu

Department of Mechanical Engineering,  
University of Texas at Austin,  
Austin, TX 78712

*This paper introduces research leading to a computer-aided design tool in which engineering designers can test various design concepts (topologies) in an environment equipped to automatically model the dynamics and conveniently optimize the specified components (given the evaluation criteria defined by human designers). A component repository is developed to store not only the component dynamics models, but also other information such as typical component design constraints and physical constitutive laws. In this paper, automated modeling of design configurations is introduced through a design representation called a conceptual dynamics graph (CD graph) and generic models of various components. CD graphs contain the information on how physical components as well as their generic models are topologically connected. A generic component model can accommodate various types of coupling between this component and its environment. This paper also discusses a systematic approach to automatically prepare a mechatronic design problem for the use of optimization to tune the parameters for optimum dynamics. Since genetic algorithms are used for this optimization, this preparation decodes and encodes proper design variables into design genotypes while taking into consideration the design constraints and physical constitutive laws. [DOI: 10.1115/1.2885180]*

## 1 Introduction

Conceptual design of dynamic systems is one of the most demanding and yet poorly understood aspects of mechanical engineering. It requires the designer to assemble a multitude of parts selected from a large discrete set of components to perform a function. The components contain physical properties and parameters that may have continuous or discrete values that must be selected for a final design. At the same time, the designer must satisfy a variety of constraints (limits, inequalities, etc.) on the overall design as well as for the different components while reconciling conflicting cost functions. In a typical design scenario, a design problem will call for the engineers to satisfy specifications for various qualities of an electromechanical device such as constraints on power consumption, desired bandwidth and accuracy bounds, and stress failure avoidance. Such mechatronic devices represent a confluence of various experts' efforts, including mechanical, electrical, control, computer, and manufacturing engineers.

During the initial conceptual design phase, detailed values about each component are not known, but rather a simple connectivity of components is described. Typically, the configuration, is divided into subsystems, and different engineers (teams) are assigned to develop each subsystem of the device. Each team receives newly specified requirements that allow the finished subsystems to come together into a complete product. Often, such "over-the-wall" design practice can lead to final designs that are cumbersome or lack robustness in meeting all the originally specified design requirements. It is only through various redesigns that a design team is able to see the subtleties in how various subsystems interact and how the individual components can be optimized to arrive at a more elegant solution to the original design problem.

As a solution to this typical scenario, a computer-aided design tool is envisioned, as shown in Fig. 1. Accompanied by the design

specifications, engineering designers would draft an entire configuration or topology of their concept perhaps from a function structure [1] as in the upper-left block of Fig. 1 by simply "dragging and dropping" elements from a component repository. The details of the coupling between any two components are captured by a multidomain design representation referred to as conceptual dynamics graph (CD graph). There are various ways for two components to be coupled, and the coupling format could be decided intelligently by the automatic detection of connection compatibility between components, which can be further compensated with designers' instructions if necessary. Through the "Simplifying CD-Graph" step, both the computational system and the designer gain a better understanding of the system by extracting the active partial configuration that contributes directly to the system dynamics [2].

The next block, "Generating Dynamics Model," aims at obtaining the system dynamics model based on a conceptual design represented by a CD Graph. Once the system model becomes available, the transformation from the system model (e.g., state space representation) to another desired representation (e.g., transfer function) can be automated. Since design goals are clearly defined prior to any computational design effort, so are the strategies that are used to evaluate the design's performance. The design performance, as design outputs, is determined by the design inputs and the system transfer function. Hence, the conclusion comes that the design evaluation function (or the objective function) can be obtained automatically from a CD graph. In this research, automated evaluation is demonstrated through automated bond graph (BG) modeling and model transformation according to the goals defined by designers.

The last step, given the evaluation function, is to automatically invoke an optimization process to determine the choices for the design variables in each of the components pulled from the repository into the configuration. Design variables can be decision variables (e.g., length of a bar component) or dependent variables (e.g., stiffness of a spring component). This paper discusses a systematic method to automatically prepare a design problem for the application of optimization using genetic algorithm (GA). This preparation sets up the genotype representation for a design by

Contributed by the Design Theory and Methodology Committee of ASME for publication in the JOURNAL OF MECHANICAL DESIGN. Manuscript received November 19, 2006; final manuscript received October 3, 2007; published online March 19, 2008. Review conducted by Janet K. Allen.

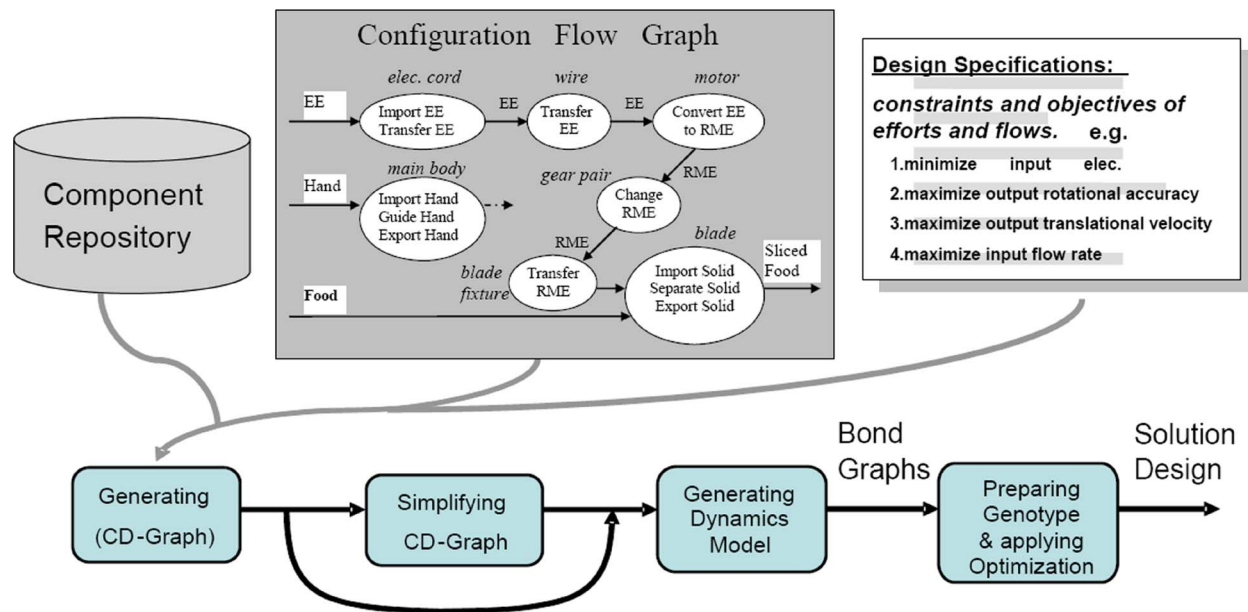


Fig. 1 Graph depicting the assistive design tool introduced in this paper

encoding design variables while taking into consideration the design constraints and physical constitutive laws that were prespecified in the component repository.

## 2 Related Work

A requirement in the design of dynamic systems is the availability of dynamics models of individual components and an effective mechanism to ensemble them into a consistent model. In order to allow for an automated modeling of electromechanical systems, a detailed library of components must be accessible for reference to build aggregate dynamics models. Given the large number of original equipment manufacturer (OEM) parts and custom components that can be used in a given configuration, it is a difficult task to construct a library that is useful to designers. Fortunately, the construction of such a component repository is well under way [3,4]. Researchers at The University of Texas at Austin, University of Missouri-Rolla, and other institutions have been building the details of this repository by dissecting and recording each individual component in a given artifact [5].

The BG formulation is used for the dynamics model development because it facilitates the integration of component/subsystem models, provides the user with physical insight, and allows easy manipulation of models. A “word BG” [6] is a less detailed, higher-level representation, where major subsystems are represented by words. Once a BG is created for each subsystem, the system BG model of the drive chain can be generated using the interconnections (power bonds). However, without a detailed design description, engineering assumptions (such as the coupling type between the shafts and the belt drive) have to be used, especially when dealing with designs of multiple domains (e.g., 2D motion with three domains of  $x$ ,  $y$ , and  $\theta_z$  instead of only one rotation domain). In modeling practices, word BGs are more used as an assistive strategy to modularize the system for modeling instead of a systematic approach for automated modeling of design configurations. Systematic introduction of a BG can be found in Refs. [6–8]. Some details about BG will be introduced in Sec. 4, which introduces examples of generic models.

Similar to the research in automated modeling is the research in automated design of electromechanical configurations. The A-design system developed by the Campbell et al. [9] automated the designing of configurations by employing design agents that add elements to the design from a component library until the

design meets the specified qualitative goals. This work as well as earlier approaches by Finger and Rinderle [10], Welch and Dixon [11], and Ulrich and Seering [12] acknowledged the possible use of BGs to capture the behavior of various components. Such approaches, however, have not made the distinction between the functional or purpose-driven reasoning used by designers to create a configuration and the behavioral or dynamic representation extracted in analyzing completed configurations. In the current repository efforts, the former description of function has been captured. The addition of behavioral models and design constraints within this research seeks to further provide engineering designers with a resourceful “computational design partner” to evaluate design concepts represented as CD graphs.

GAs are biologically motivated adaptive systems, which are based on the principles of natural selection and genetic recombination. In the standard GA, candidate solutions are encoded as fixed length vectors. The initial group of potential solutions is chosen randomly. The subsequent generation is created through a process of selection, recombination, and mutation. Although the solutions with high fitness values have a higher probability of selection for recombination than those with low fitness values, they are not guaranteed to appear in the next generation. A good overview of GAs can be found in Refs. [13,14].

A number of other approaches explored the design automation of dynamic systems. Using a BG formalism, Tay et al. [15] and Seo et al. [16] combined a BG method with a GA or with genetic programming (GP), respectively, to construct BG dynamic system models from BG embryos. However, there is no direct mapping from the BG elements to physical components in an electromechanical system design. In our approach, we start from the physical topology design upon which component BG models are aggregated automatically to obtain a system model.

Although some commercial software packages such as ADAMS [17] and DYMOLA [18] support automated modeling of physical designs configurations, there is limited flexibility in obtaining analytical models in a user-desired format to integrate with optimization that requires specific evaluation routines. In ADAMS, the dynamics equations are generated numerically from the geometric description of the mechanism [19]. In DYMOLA, modeling knowledge stored in libraries can be reused to generate system models using an object-oriented modeling language MODELICA. Further approaches for system modeling include block diagram [20], lin-

**Table 1 Coupling types**

Couple type	Index	Modeling Interpretation
Decoupling	C0	Disconnect ports
Plain coupling	C1	Connect directly
Impedance coupling	C2	Add impedance element
Discrete plain coupling	C1*	Add switch element
Discrete impedance coupling	C2*	Add switch and impedance elements

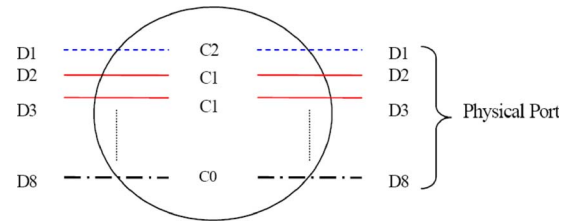
ear graph [21], etc. However, in this research, BG, are used for system modeling that provide more physical insights than aforementioned approaches due to the energy-based nature of BG elements [7]. Further, the CD graph proposed in this paper as a highly conceptual design representation provides an easier grip for design engineers to set off not only automated modeling of a design but also automated design evaluation by combining GA-based parameter optimization.

Some existing BG softwares such as CAMP-G [22] and SIMBOLS [23] allow users to create BGs and derive system equations, but not emphasize the concept of generic models to facilitate automatic modeling of design configurations. No explicit documents that provide detailed generic models for various components have been found like in this paper although it is possible that models with similar functions might have been used within some applications.

### 3 Conceptual Dynamics Graph

An appropriate model can be developed for a design configuration only if this configuration can be described in a perceivable manner. Various physical couplers can have the same functionality although they may have very different geometries, assembly methods, numbers of parts, etc. For example, screws and glue can both be used to assemble structurally two components together although in different manners. Further, when we say that an automobile's engine is connected to the automobile's chassis, we are being vague. In fact, it is more precise to say that the engine mounted on the chassis is fixed to pick up points on the engine block by means of some fasteners. That is, the relation between the engine and the chassis is most concretely described in terms of parts (or even features of parts) of the overall assembly. This is essentially a structural view of the couplings. During the conceptual design stage, the information generally needed to specify the couplings is functional in nature, rather than structural: The function of the connection between an automobile engine and a chassis is to secure the engine to the automobile's main structural support, transmit forces developed by the engine to that structure, and ensure that other connections (e.g., that between the engine and the transmission) are maintained. In this situation, a virtual coupler (VC) is designed to capture the coupling's functional features instead of describing the structure detail.

**Virtual Coupler.** Each mechanical connection can be decomposed into couplings of six domains, which are three translational ( $x, y, z$ ) and three rotational  $\{\theta_x, \theta_y, \theta_z\}$ . With the addition of electrical and hydraulic domains, a total of eight are created (labeled D1–D8). The coupling type of a specific domain can also be any of the three basic coupling types (Table 1), which are C0 (decoupling), C1 (plain coupling), and C2 (impedance coupling). Here, plain coupling means that the coupled ends at a certain domain share the same generalized flow (velocity for mechanical components). Impedance coupling implies that there exists an impedance element in the corresponding domain between the coupled two ends, which could be a resistance due to the relative motion of the two ends (e.g., the lubrication fluid of journal bearing in mechanical domains) or compliance due to the energy storage of the in-between material. Although not realized thus far, the coupling types could be augmented to account for other situations



**Fig. 2 A VC specifies how the interactions between components occur. It provides multidomain information (D1–D8) with certain coupling types (C0–C2) at each domain between the connected components.**

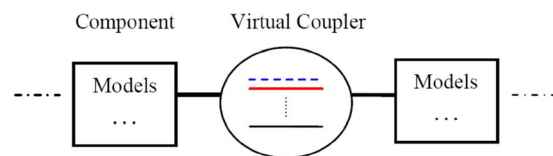
such as the “switch” or “ratchet” effect for a noncontinuous coupling (C1\* and C2\*). A VC specifies how the multidimensional interactions between components occur as depicted in Fig. 2.

A graph structure (Fig. 3) composed of VCs and components is called CD graph. Based on the conceptual dynamics (e.g., degrees of freedom or number of states), a qualitative evaluation method of design candidates is under investigation to search for the potentially feasible designs, which, in fact, guide the generation process of valid CD graphs.

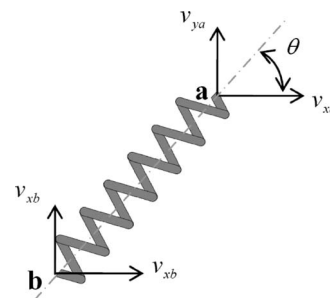
### 4 Bond Graph Model Generation

**4.1 Generic Models.** For the same component, dynamics models vary, corresponding to different forms of coupling with its surroundings, which are especially true for mechanical components due to the factors of gravity and multidimensions in the coordinate frame. To achieve the goal of modeling automation, generic models are created in this research for each component to fit within its generic surroundings to facilitate automated dynamics modeling. In this section, a number of examples of generic port-based component models are demonstrated, most of which will be used in the later case study of weighing machine design.

**4.1.1 Example 1: Generic Model of a Spring.** The spring of Fig. 4, moving in the plane, is a standard component of mechanical systems. A translational spring within a design configuration can be simply modeled as a linear capacitance if the motion of its two ends remains along the same line. However, if the spring exhibits motions that are planar (2D) or even out of plane (3D), a



**Fig. 3 CD graph**



**Fig. 4 A spring with port a and port b**



more complicated model is needed.

The two locations on the spring are the two tips  $a$  and  $b$ , each of which can be used as a port to connect to another component. The spring thus asserts a force against the components connected through these two ports. Motion is considered with respect to an absolute coordinate system:  $v_{xa}$  and  $v_{ya}$  are the components of the velocities of tip (port)  $a$  with respect to this coordinate system;  $v_{xb}$  and  $v_{yb}$  are the components of the velocities of tip  $b$ . The distance from tip  $a$  to tip  $b$  is represented as  $l$ , and with the distance at zero spring load as  $l_0$ .

The kinematics of the spring are expressed by the equations

$$\begin{aligned} \dot{l}_x(t) &= v_{xa} - v_{xb} \\ \dot{l}_y(t) &= v_{ya} - v_{yb} \end{aligned} \quad (1)$$

where  $\dot{l}_x(t)$  and  $\dot{l}_y(t)$  are the rates of length change along the  $x$  and  $y$  directions,

$$\begin{aligned} l_x &= \int_0^{t'} \dot{l}_x(t) dt + l_{x0} \\ l_y &= \int_0^{t'} \dot{l}_y(t) dt + l_{y0} \end{aligned} \quad (2)$$

where  $l_x$  and  $l_y$  are the spring length on the  $x$  and  $y$  directions with  $l_{x0}$  and  $l_{y0}$  as the initial conditions. The position angle of the spring relative to the reference frame can be obtained by

$$\dot{\theta} = \arctan\left(\frac{l_y}{l_x}\right) \quad (3)$$

$$\theta = \int_0^{t'} \dot{\theta}(t) dt + \theta_0 \quad (4)$$

Further,

$$\begin{aligned} \dot{l}_{x\theta} &= \dot{l}_x \cos(\theta) \\ \dot{l}_{y\theta} &= \dot{l}_y \sin(\theta) \end{aligned} \quad (5)$$

where  $\dot{l}_{x\theta}$  and  $\dot{l}_{y\theta}$  are the projections of  $\dot{l}_x$  and  $\dot{l}_y$  along the spring axis. The addition of  $\dot{l}_{x\theta}$  and  $\dot{l}_{y\theta}$  gives the rate of change of the spring length as follows:

$$\dot{l} = \dot{l}_{x\theta} + \dot{l}_{y\theta} \quad (6)$$

The dynamics equations is simply

$$\Delta f = f_c \left( \int_0^{t'} \dot{l}(t) dt + l_0 \right) \quad (7)$$

where  $\Delta f$  is the net force acting between ports  $a$  and  $b$ . If we assume that this is an ideal linear spring with  $C$  as the spring compliance, then

$$\Delta f = \frac{\int_0^{t'} \dot{l}(t) dt + l_0}{C} \quad (8)$$

The corresponding BG appears in Fig. 5 following the notions of model transformation tool (MTT) [24]. MTT is an open-source software to transform dynamics models from BG representation to another representation such as state space differential equations or transfer function. In Fig. 5, the source sensors (SS) indicate a spring's domain interface to other components; for example, SS:[ $x_a$ ] is the interface of tip (port)  $a$  at the  $x$  domain.

**4.1.2 Example 2: Generic Model of a Bar.** A two dimensional bar can have a translational motion or a rotational motion or a

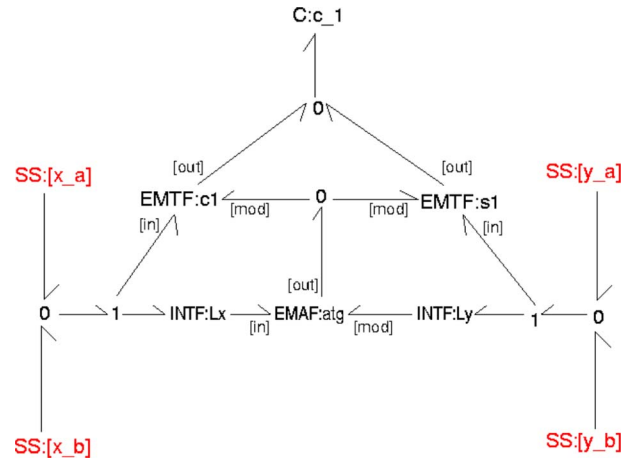


Fig. 5 Generic model of a two-port spring

combination of both. A generic model as in Fig. 6 (extended from the model originally developed in MTT [24]) accommodates these various modes of motion. Ports  $a$  and  $c$  are the two ends of the bar, and port  $b$  is anywhere in the middle of the bar. Each port of the bar ( $a$ ,  $b$ , and  $c$ ) has three domains, translational  $x$  and  $y$  and rotational  $\theta_z$  with each domain interface represented using a SS. Accordingly, the bar's inertia has an effect on each domain, represented as  $m_x$  and  $m_y$  for the linear inertia at the  $x$  and  $y$  directions, and  $J$  for the moment of inertia with respect to port  $b$ .

The integrated transformer (INTF) is a BG submodule that takes flow (angular velocity for the bar example) as an input and gives the flow integration as an output (the body position angle "theta" relative to the reference frame). An effort modulated transformer (EMTF) represents a transformer that is modulated by an effort variable, which, in this example, is the output of the INTF component (angle position). C1, c2, s1, and s2 are all instantiations of the modulating function, such as "cos(theta)" or "sin(theta)." The marks [mod], [in], and [out] are used to differentiate the bonds connected to each EMTF transformer, which can be recognized by a BG interpreter in the following model transformation process. This BG can also be in a script representation with the same information captured. Note that the model of a mechanical lever

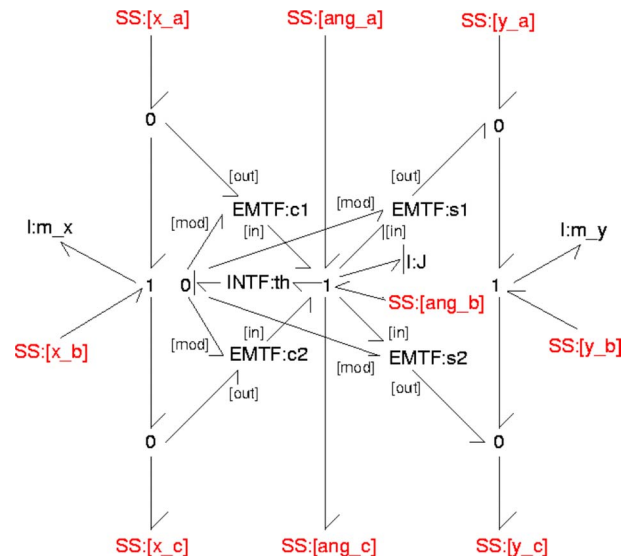


Fig. 6 Generic model of a 2D three-port bar

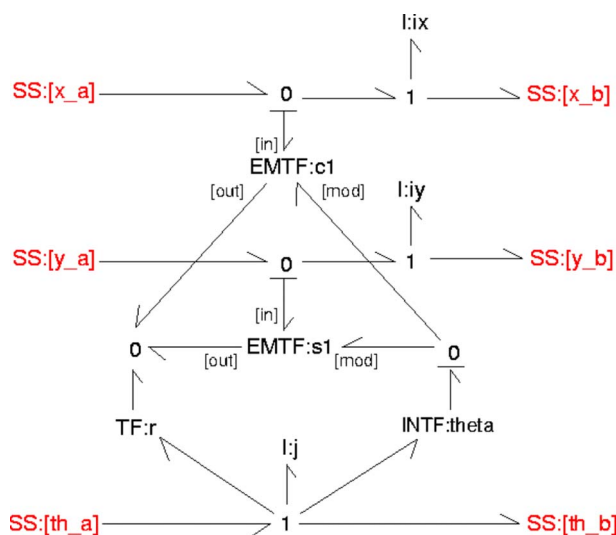


Fig. 7 A 2D generic model of a wheel (gear) with two ports

could be easily obtained from this bar model by pinning any of the three ports to a fulcrum. Further, more ports can be added in the middle of the bar in the same manner as the end ports.

**4.1.3 Example 4: Generic Model of a Wheel (Gear).** A wheel represented in only two dimensions only has a single rolling motion. The relative velocity between the outer contour and the wheel center (or the absolute velocity of the disk's center if there is no slipping between the wheel and the ground) is the product of the rolling angular velocity and the radius of the wheel,

$$v = w^*r \quad (9)$$

Figure 7 shows a 2D generic wheel BG model where  $TF:r$  is used to capture the relation (Eq. (9)) between the linear velocity and the angular velocity. The model has two ports: One (port  $a$ ) is the center of the wheel and the other (port  $b$ ) is the outer contour. The port (domain) can be, in practice, rigidly coupled, slipping, or decoupled with its connecting component (an axle for port  $a$  and ground for port  $b$ ).

There is a similar relation for the wheel moving in a 3D space, which can be derived if a body-fixed coordinate frame is defined where

$$\mathbf{v} = \mathbf{w} \times \mathbf{r} \quad (10)$$

Equation (10) shows how the angular velocity vector ( $\mathbf{w}$ ) can be related to the linear velocity vector ( $\mathbf{v}$ ) in the body-fixed coordinate frame originated at the center point of the wheel. It is obvious that relative to the body-fixed coordinates,  $\mathbf{r}$  is independent of the pitch angle  $\theta$  and the yaw angle  $\psi$ , but dependent on  $\phi$ , the rolling angle along the  $x$  axis, which is

$$\mathbf{r} = (0, R \sin \phi, R \cos \phi) \quad (11)$$

Following the rule for the vector product, we have

$$\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} 0 & R \cos \phi & -R \sin \phi \\ -R \cos \phi & 0 & 0 \\ R \sin \phi & 0 & 0 \end{bmatrix} \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} \quad (12)$$

A general 3D rigid-body dynamics model and its detailed derivation can be found in Ref. [6], based on which a 3D wheel model (Fig. 8) can be obtained by adding a constraint specific to the wheel.

This constraint is the relation between the linear velocity and the angular velocity of the disk center relative to the body-fixed coordinates, as captured in Eq. (12). Similar to the 2D wheel model in Fig. 7, the 3D model has the same number of ports and

same manner of interfacing with its environment, though the 3D model has six domains instead of three domains for each port and is more complicated.

The level of a generic component model to be used in the modeling of a design configuration, which could be 3D (six domains), 2D (three domains), or even 1D (one domain), simply depends on the design's potential range of motion. For example, a 3D model needs to be used if the component is designed to move in a 3D path (e.g., a wheel on a sports utility vehicle or the Euler disk problem [25]), but a 2D model can be used for simplicity if it is to move in a 2D plane (e.g., a planet gear or a train wheel that travels along a rather straight line), or the model can be further simplified to 1D for some cases (e.g., a fly wheel used to store kinetic energy).

The range of motion can be qualitatively determined from the design's CD graph. If the coupling types at all those domains that lead to a higher dimension are rigidly coupled with ground, then a lower dimension component model can be used. The activity of the nonmechanical domains can also be determined from the CD graph simply by checking the coupling types of these domains with its environment, which tells how complex a model needs to be in terms of degrees of freedom or number of domains. Further granularity may be pursued to have specific models for more specific application situations. For example, a shift gear used in the automobile transmission has two active domains including one translational domain and one rotational domain along the same axis.

Note that for all the demonstrated examples, coordinate the transformation between component body frame and reference frame is integrated into the BG representation of component models. This mechanism provides a fundamental convenience to modeling the VCs and thus to generating the system model.

**4.2 System Model.** BGs provide a unified "model" representation, which adopts generalized components such as capacitance ( $C$ ) and inertia ( $I$ ) that are independent of a specific domain. Based on BGs, the CD graph provides a "design" representation that facilitates the automated modeling of multidomain design configurations through a mapping between a system's generic model configuration and physical topology configuration. This mapping also indicates that when a physical system is represented in a Cartesian coordinate frame, so can the model(s) of the physical system.

A schematic 2D crank-slider design structure is shown in Fig. 9(a) as an illustrative example of system level model generation. Figure 9(b) shows the CD graph of this design configuration. Each 2D port of the components and VCs has three domains ( $x, y, \theta_z$ ), with all the other mechanical domains plain coupled and nonmechanical domains decoupled. Figures 9(c)–9(e) represent the BG models for VC2 and VC3 ( $C1, C1, C2$ ), VC4 ( $C1, C1, C0$ ), and VC5 ( $C2, C1, C1$ ), respectively. VCs 2 and 3 has plain coupling for  $x$  and  $y$  domains (marked as  $C1$ ) and impedance couple for  $\theta_z$  domain (marked as  $C2$ ). VC4 is different from VCs 2 and 3 since its  $\theta_z$  domain is decoupled by the assumption that there is no rotational friction (marked as  $C0$ ). VC5 has its  $x$  domain defined as impedance coupled (marked as  $C2$ ).

Note in the expression  $S_x(0,1)$  in (d), where  $S_x$  is the model interface corresponding to the  $x$  domain of the physical port (following the definition in MTT [24]). Within the parentheses following  $S_x$ , the first binary digit represents effort (force in mechanical), and the second one represents flow (speed in mechanical).  $S_x(0,1)$  means that at the  $x$  domain, effort variable is grounded to zero, while the  $x$  domain flow variable is determined by dynamics of the overall system. Visually, in the system's bond graph,  $S(0,1)$  works as a "plug" (or a single domain component) in a domain, which asserts a free end (zero effort) to the corresponding domain of the connected component. For a 2D pendulum example, the free end is coupled with a VC modeled simply by three  $S(1,0)$  plugs since this end is free of restriction at any of

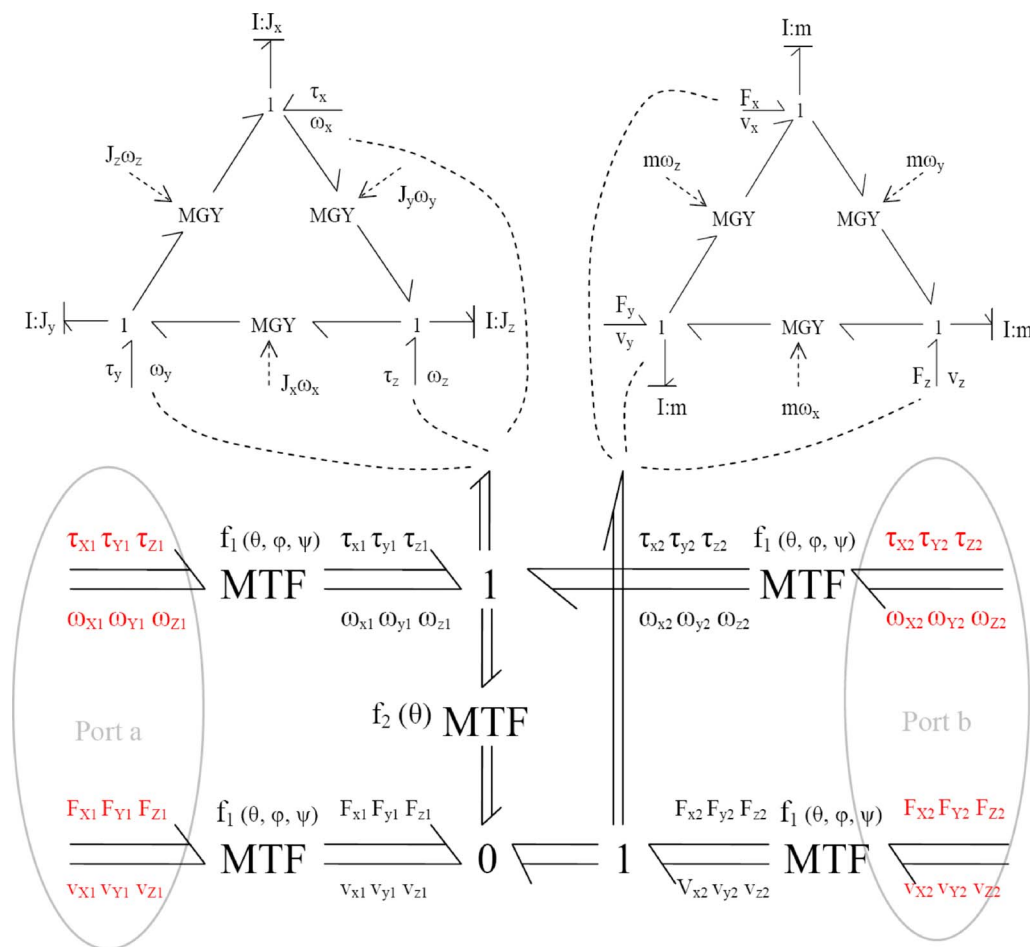


Fig. 8 3D generic model of a wheel (gear) with two ports

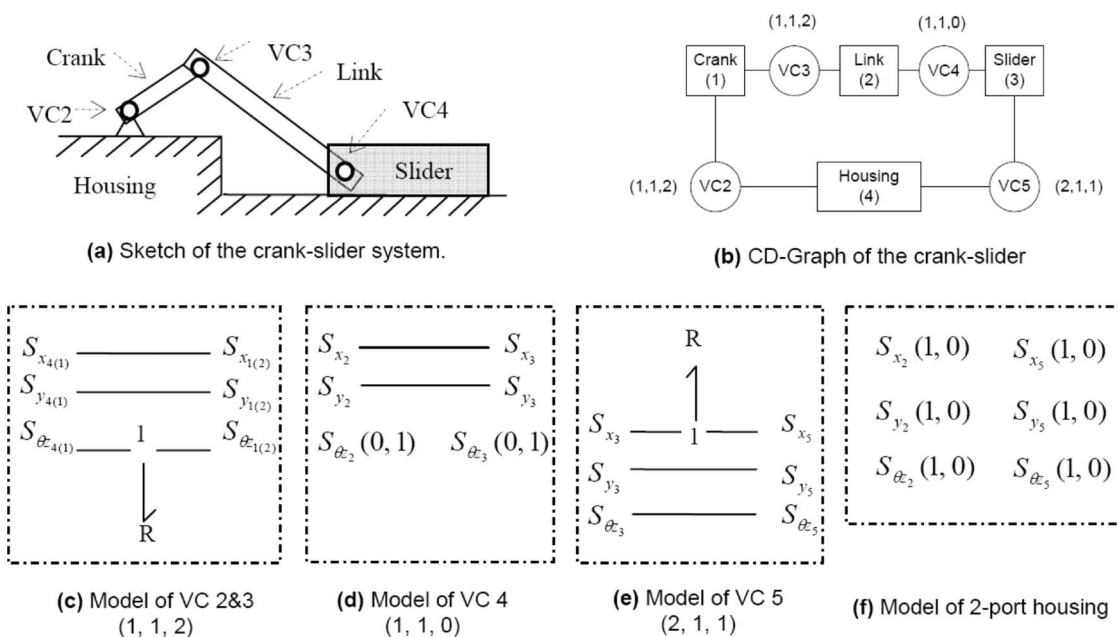


Fig. 9 The system model generation of a 2D crank-and-slider system can be done by connecting the ports in a domain-to-domain manner. The schematic geometric expression of the simplified crank-slider conceptual design is shown in (a) and its CD graph in (b). (c)–(e) represent the BG models for VCs.

the three domains.

Further, a two-port ground model is shown in (f), although it can be reduced to one port or expanded to as many ports as needed. For this ground model, all the domains have zero flow but an arbitrary value for effort which is determined by the input, so the  $x$  domain of the ground component can be marked as  $S_x(1,0)$ , likewise for the other two domains. Visually,  $S(1,0)$  works as a plug in a domain, which asserts a ground (zero flow) to the corresponding domain of the connected component. This feature becomes useful when a simple model of a component (e.g., a 2D model while the other domains are grounded) is connected to a complicated model of another component (e.g., a 3D model). Any unconnected domain of the VC between the two models (one with six domains, another with three domains) can be connected to (or plugged by) a single domain component that is simply modeled as  $S(1,0)$ .

Given the general behavior models of all the components (both crank and link are considered as a bar model, as shown in Fig. 6) and all the VC models, a system level model (Fig. 11) can be created by simply connecting the components and couplers intuitively in a domain-to-domain manner (i.e.,  $x$  to  $x$ ,  $y$  to  $y$ ).

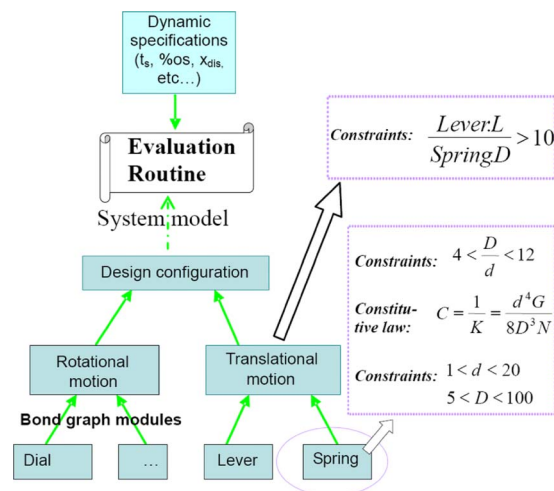
After the system bond graph model is generated, ground information can be applied to remove the BG elements with no energy effect on the rest of the system (such as the dangling elements). There are other energy-based BG model reduction algorithms [26] but all with the condition that the detailed design parameters are known, which is not true for our application at the conceptual design stage.

**4.3 Automated Preparation for Optimization.** After a system dynamics model (BG) is generated, the model transformation from this graphical representation to other representations such as state space or transfer function can be automated through the MTT. Design goals (e.g., settling time for a step input, overshoot, bandwidth, etc.) are often as clearly defined as possible by engineers prior to any design effort, which suggests that the evaluation routine for designs can be obtained by representing the design goals as a function of system output variables, which are derivable from the system inputs and dynamics model. Based on this evaluation routine, an optimization task can be invoked to determine the choices for the design variables in each of the components pulled from the repository into the configuration. Design variables can be decisive variables (e.g., length of a bar component) or dependent variables (e.g., stiffness of a spring component).

The dynamics models built will likely include significant nonlinearities that result in a multimodal design space. Since gradient-based optimization methods will most likely fail in such scenarios, GA [14]—a stochastic-based approach—is adopted to find the best set of parameters. The major appeal is due to its attributes of robustness and global searching. In a GA, the design solution candidates (phenotype) are encoded in a list representation called a genotype. Although GA based optimization is not the emphasis of this research, a unique and necessary task is to, given the evaluation routine, automatically prepare a design problem for the application of optimization using GA. This preparation sets up the genotype representation for a design by encoding design variables while taking into consideration the design constraints and physical constitutive laws that were prespecified in the component repository. An example of weighing machine design is used to showcase this research through a detailed explanation.

## 5 Case Study of a Weighing Machine Design

Figure 10 shows an example structurally of the optimization problem for a weighing machine design with its CD graph shown in Fig. 12. The weighing machine conceptual design takes weight as an input to a footpad that is rigidly coupled with a lever [9]. A spring is used to support the lever (a bar) whose translational motion at the free end drives a rack pinion (a bar and a gear) to generate rotational motion, which passes through a shaft and then



**Fig. 10 Optimization for a weighing machine design. Samples of design constraints and constitutive laws are listed at different levels.**

displayed by a gauge (dial) displacement. Among the specifications of Fig. 11 are the dynamics goals such as the oscillation of the output gauge is expected to settle down within 3 s and the percent overshoot is expected to be less than 20%, etc. As a result, the optimization is faced with multiple objective functions as will often be the case in such problems. The system model of this design can be automatically generated from the CD graph and then converted into desired formats (e.g., transfer functions between multiple inputs and outputs), which is invoked by the evaluation routine that reflect the design specifications. The computational choice of component parameters should comply with the “rules of thumb” that are followed by the design engineers. These rules may be cast as equality or inequality constraints for the optimization problem. For instance, as shown in Fig. 11, the ratio between spring coil diameter ( $D$ ) and wire diameter ( $d$ ) is usually expected to be greater than 4 to allow for easier fabrication and less than 12 to avoid tangling. Also, the constraints of system level (relation between variables from different components) can be predefined or open to designer’s specification. As shown in Fig. 11, a design heuristic poses a requirement that the length of lever is at least ten times larger than the coil diameter of the spring.

The BG system model is represented with the attributes of the components and couplers, such as the capacitance element  $C$  and resistance element  $R$ , which are intermediate variables from a perspective of product design. It is our goal to relate the generalized BG attributes of components to basic physical design variables, such as dimensions, materials, etc. Constitutive laws or mappings between the intermediate and basic variables, which are defined in the “behavioral model” section of the component representation, are needed. For example, Fig. 11 defines the spring stiffness as a function of basic design parameters including the number of coils, coil diameter, etc. In practice, the physical design variables are constrained to certain boundaries, such as the dimensional range, or to some limited discrete values, such as the material properties. These constraints are defined within the “design constraints” section of the component representation. Figure 11 shows examples of such boundaries for wire diameter and coil diameter for the spring design.

**5.1 Preparing Design Genotype.** A task prior to applying GA for design optimization is to form the genotype automatically, which entails deriving the proper set of design variables and their ranges. In this research, component design information such as constitutive laws and constraints are captured through a hierarchi-



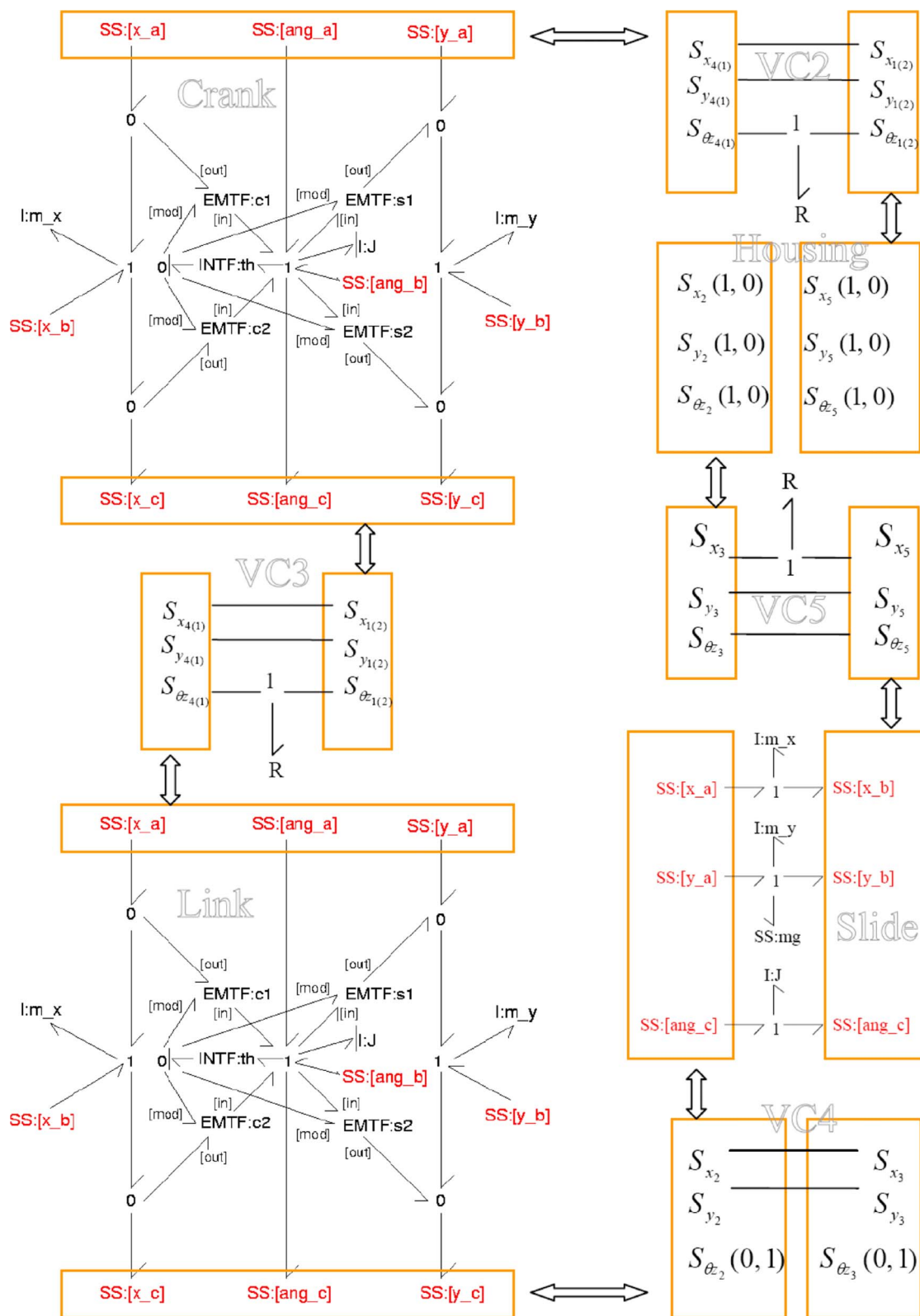


Fig. 11 The system BG model generated from the CD graph

cal structure (e.g., Figs. 13(a) and 13(b)) with the top node marked as component name, middle nodes as intermediate design variables (nested circles), and leaf nodes as basic design variables (single circles). In the spring example (a), the constraint (dia-

mond) and the constitutive relation correspond to the upper and lower limits on wire diameter and the constitutive stiffness equation, and the lever example (b) captures the constitutive relations between  $J$  and  $I, L_a, L_b$  and between  $I$  and  $L_a, L_b, A, \rho$ .



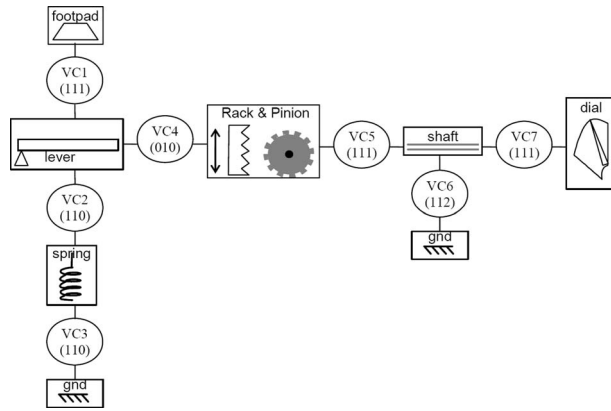


Fig. 12 CD graph of a weighing machine design

Note that any physical parameter can be a design variable and be subject to a tuning process. However, prior to the beginning of the computational design process, the designer has the option to fix the values of some variables (e.g., variable  $G$  marked in solid line circle(s)) and/or to set up boundaries for some other design variables (e.g., variable  $N$  in (a)). If an intermediate variable is chosen as a design variable (e.g., variable  $I$  in (b)), its single-parented children (such as variable  $A$ ) will not be chosen as design variables to prevent conflicts. The dashed circles (single or nested) are the set of design variables to be encoded in the genotype as partially shown in Fig. 13(c).

**5.2 Fitness Function.** Through automated modeling and model transformation, state space equations (Fig. 14) are generated automatically from the CD graph for the weighing machine design by invoking the MTTs. As is shown, there are seven integrative states and six derivative states. Two outputs were defined as functions of two states prior to the computational design ef-

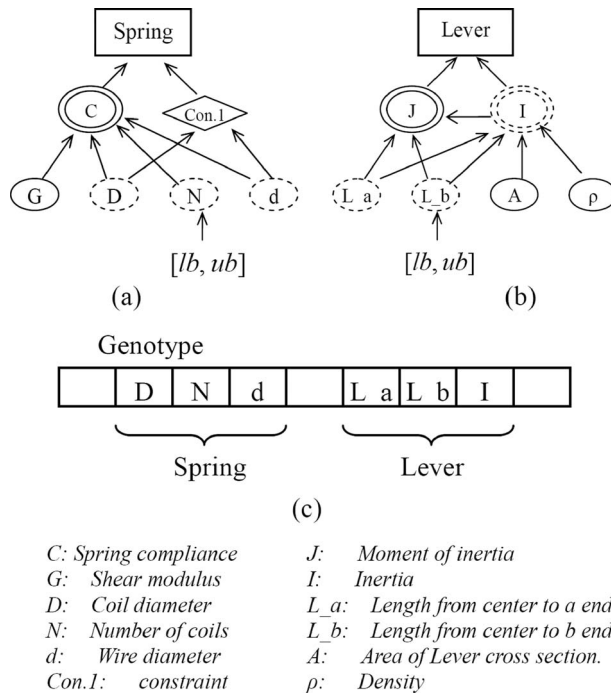


Fig. 13 Automatic construction of genotype representation (c) based on the hierarchical representation of component design rules (a) and (b)

$$\begin{aligned} \dot{x}_1 &= \frac{x_4}{sh1_c} \\ \dot{x}_2 &= \frac{(\cos(x_5)le1_jle1_{ia}z_5sh1_csp1_c - \sin(x_5)^2le1_{ia}^2x_2rp1_r p1_{if}^2sh1_csp1_c)}{le1_j} \\ &\dots \\ \dot{x}_7 &= \frac{(\sin(x_5)le1_{ia}x_2)}{le1_j} \\ z_1 &= \frac{(\sin(x_5)le1_{ia}x_2we1_i)}{le1_j} \\ &\dots \\ z_6 &= \frac{(\sin(x_5)le1_{ia}le1_mx_2)}{le1_j} \end{aligned} \quad \begin{aligned} y_1 &= x_6 \\ y_2 &= x_7 \end{aligned}$$

Fig. 14 Equations generated from the CD graph of the weighing machine by automated modeling based on modularized BG models. There are two outputs defined at the beginning of the design process,  $y_1$  (the footpad displacement) and  $y_2$  (the angle of the dial rotation).

forts,  $y_1$  (footpad displacement) and  $y_2$  (angle of dial rotation). In terms of these two outputs, there are three design targets as shown with the dashed lines in Fig. 15. With the input of 100 lb at the footpad, they are as follows: (1) dial rotation equals to 1 rad:  $f_1(y_1)$ ; (2) the footpad displacement is less than 2 cm (as close to 0 as possible):  $f_2(y_2)$ ; (3) dial settles down within 3 s:  $f_3(y_1)$ . The three cost functions are obtained from numerical calculations over a number of distributed sample points. The overall evaluation of a design is simply the weighted sum of these three cost functions. Note that the objective is to “minimize” the overall cost to obtain the best design fitness. The green dotted line and the blue solid line (Fig. 15) are the footpad and dial responses, respectively, of the best design.

Details of the shaft and spring design are shown in Table 2. The values in the column “Value assigned” are assigned to the corresponding properties (such as material density of the shaft) by designer’s preference, while leaving other properties to be determined by the subsequent computational design efforts. Through the aforementioned strategies on preparing genotype, a proper set of properties were encoded into the genotype as design variables, and the values of these design variables are shown in the column “Value designed.” The design constraints and constitutive laws are listed in the next column and used to either validate the values designed or calculate other intermediate properties. To reduce the

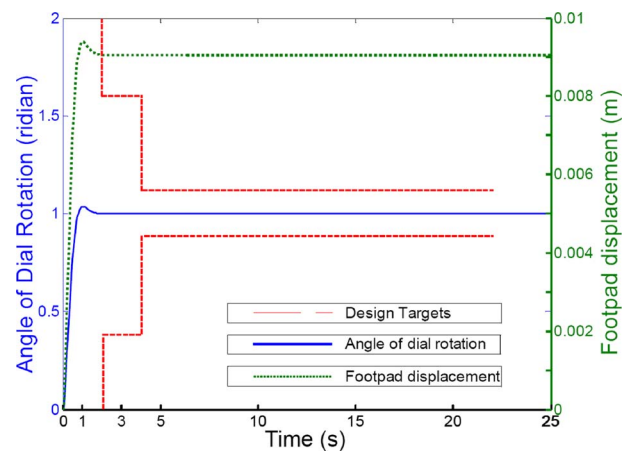


Fig. 15 Response of the best design obtained by GA optimization for weighing machine design. The green dot line and the blue solid line are the footpad and dial responses, respectively.

**Table 2 The parameters designed for the components shaft and spring**

Component	Property	Value designed	Constitutive laws and constraints	Value assigned	Lower limit	Upper limit
Shaft	Shear modulus: $G$ Pa			$7.31E+10$		
	Density: $\rho$ (kg/m <sup>3</sup> )			$7.75E+03$		
	Rotational compliance: $C$ (1/N.m)		$=1/(G*\pi*D^4/(32*L))$ $=2.7349E-003$			
	Rotational inertia: $I$ (kg.m <sup>2</sup> )		$=\rho*\pi*LD^4/32$ $=8.7770E-007$			
	Length: $L$ (m)	$1.5038E-01$			0.05	0.5
	Diameter: $D$ (m)	$9.3570E-03$		0.002	0.02	
Spring	Ratio constraint: $R$	$9.7535E+00$	$4 < D/d < 12$		4	12
	Shear modulus: $G$ (Pa)			$7.31E+10$		
	Wire diameter: $d$ (m)		$=D/R$ $=5.1881E-003$		0.001	0.020
	Coil diameter: $D$ (m)	$5.0602E-02$			0.005	0.10
	Number of coils: $N$	$9.6691E+00$			4	20
	Compliance: $C$ (m/N)		$=(8*D^3*N)/(d^4*G)$ $=1.8932E-004$			

design space, a range for each design variable was stored in the component repository based on common applications and open to designer's modification.

**5.3 Experimental Setup.** In the implementation of this research, Matlab® is used as a programming platform and combined two existing public-domain software, MTT [24] and GAOT [27]. MTT is used to obtain system differential equations from the BG representation. GAOT is a GA package that supports real-valued GA, which is used in this research so that the parameters' values evolved by genetic operations remain within their corresponding range limits.

## 6 Discussion

This paper introduces research leading to a computer-aided design tool in which engineering designers can conveniently test design concepts (topologies). A design representation called CD graph is used as the interface that relates a topology of components to its system dynamics model. Based on the CD graph, a qualitative evaluation of computational generated design concepts needs to be addressed to screen out the valid design concepts for a detailed parameter design, which is currently among our ongoing research.

To automatically model a design configuration, a generic model is predefined for each component that accommodates various interactions with the environment of this component. VCs of CD graphs record the interaction format in which generic models of components are assembled topologically. The transformation between global and local coordinates is also captured in the generic models. However, further work is still needed to enrich the repository of BG generic models to make this work more powerful. Currently, generic models (of different complexities) for ten components are implemented in the system, which includes gear, wheel, motor, lever, spring, pulley, shaft, bar, rack, and generator.

Automated optimization practice is built on predefined performance targets. Some components may be specified in the to-be-finished topology (incomplete design) to carry the overall system inputs and outputs, for example, the footpad and dial in the weighing machine design. Note that it is known that design requirements evolve throughout the conceptual design stage [28]. The method proposed in this paper currently does not account for evolving requirements, but rather optimizes a design for a fixed set of targets.

Some commercial software packages such as ADAMS and DY-MOLA support automated modeling. However, they have limited ability in modeling designs represented in a highly conceptual

fashion like CD graph, which provides an easier grip for design engineers to simplify design configurations by lumping together the components with the same functionality and to choose models with appropriate degrees of freedom. Further, there is limited flexibility in such software in obtaining analytical models in a desired format to integrate with optimization that requires specific evaluation routines. In this research, BGs are used for system modeling that provide more physical insights than other approaches such as block diagram or linear graph due to the energy-based nature of BG elements.

**6.1 Model Complexity.** It is in the designers' interest to simplify both a design configuration and its dynamics model. In a design configuration, some components contribute to the system dynamics directly, while some other components do not since they are rigidly coupled and share the same function with the aforementioned components or are structural components that do not consume or provide energy. Further details on how to simplify design configurations are introduced in [2]. By reducing the number of components in a design configuration to only those that impact the system's dynamics, both designers and the computational system can more easily manage the conceptual system.

Dynamics models are an abstract representation of the behavior of a system as it interacts with its environment. As demonstrated in the previous section, a component can have models of different complexities in terms of degrees of freedom according to the environments the component interacts with. However, it is important to note another type of complexity, which is the number of modes modeled for a component. A dynamic mode refers to the degrees of freedom of the component as well as the complexity of how it transfers energy among its ports. For a certain context (such as particular inputs, choices of design parameters and neighboring components), some modes are dominant while other modes are negligible. The priority of modes can vary significantly when the context changes. Under the condition that all the parameters of a design configuration are known before hand, several authors [29,30] have looked at this problem and investigated algorithms to quantitatively deduce models with a proper number of modes captured to meet the modeling requirements in frequency domain. In this research, it is not known in advance which modes can be ignored since the design variables dictating the modes have yet to be determined.

## 7 Conclusions

In this paper, a conceptual design instantiation procedure composed of automated modeling and computer-aided optimization

was elaborated, which is leading to a computer-aided design tool in which engineering designers can conveniently test various design concepts to best meet the dynamics specifications of design problems. The input to the system is a graph of components (CD graph) where the components' design variables are to be determined by the subsequent optimization based on the analytical system model generated automatically.

In this research, automated BG modeling was introduced, which relieves the burden of understanding the system dynamics of design configurations by leveraging generic models stored in a component repository. This paper also discussed a systematic approach to automatically prepare a design genotype for the application of GA optimization using the stored design heuristics to instantiate the design variables. This preparation can encode and decode proper design variables into the genotype in a sense that accounts for the existing design constraints and physical constitutive laws.

## References

- [1] Kurtoglu, T., Campbell, M. I., Gonzalez, J., Bryant, C. R., Stone, R. B., and McAdams, D. A., 2005, "Capturing Empirically Derived Design Knowledge for Creating Conceptual Design Configurations," *ASME 2005 International Design Engineering and Technical Conference and Computers and Information in Engineering Conferences*, Long Beach, CA.
- [2] Wu, Z., Campbell, M. I., and Fernandez, B. R., 2005, "A Design Representation to Support the Automatic Dynamic Evaluation of Electromechanical Designs," *Proceedings of the 2005 ASME International Mechanical Engineering Congress and Exposition*, Orlando, FL, Nov. 5–11.
- [3] Szykman, S., Racz, J., and Sriram, R., 1999, "The Representation of Function in Computer-Based Design," *Proceedings of DETC99*, Las Vegas, NV, Paper No. DETC99/DTM-8742.
- [4] 2000, *NIST Workshop on Product Representation for Next-Generation Distributed Product Development*, National Institute of Standards and Technology, Gaithersburg, MD, Nov. 30–Dec. 1.
- [5] Bohm, M., and Stone, R., 2004, "Representing Functionality to Support Reuse: Conceptual and Supporting Functions," *Proceedings of DETC'04*, Salt Lake City, UT, Paper No. DETC2004-57693.
- [6] Karnopp, D. C., Margolis, D. L., and Rosenberg, R. C., 2000, *System Dynamics: Modeling and Simulation of Mechatronic Systems*, Wiley, New York.
- [7] Beaman, J. J., and Paynter, H. M., 1995, *Modeling of Physical Systems*, University of Texas at Austin, TX.
- [8] Borutzky, W., 1999, "Bond Graph Modeling From an Object Oriented Modeling Point of View," *Simulation Practice Theory*, 7(5–6), pp. 439–461.
- [9] Campbell, M. I., Cagan, J., and Kotovsky, K., 1999, "A-Design: An Agent-Based Approach to Conceptual Design in a Dynamic Environment," *Res. Eng. Des.*, 11(3), pp. 172–192.
- [10] Finger, S., and Rinderle, J. R., 1989, "A Transformational Approach to Mechanical Design Using a Bond Graph Grammar," *Design Theory and Methodology, DTM'89*, DE Vol. 17, pp. 107–116.
- [11] Welch, R. V., and Dixon, J., 1994, "Guiding Conceptual Design Through Behavioral Reasoning," *Res. Eng. Des.*, 6, pp. 169–188.
- [12] Ulrich, K. T., and Seering, W., 1989, "Synthesis of Schematic Descriptions in Mechanical Design," *Res. Eng. Des.*, 1, pp. 3–18.
- [13] Goldberg, D. E., 1989, *Genetic Algorithms in Search, Optimization, and Machine Learning* Addison-Wesley, Reading, MA.
- [14] Holland, J., 1992, "Genetic Algorithms," *Sci. Am.*, 267, pp. 66–72.
- [15] Tay, E., Flowers, W., and Barrus, J., 1998, "Automated Generation and Analysis of Dynamic System Design," *Korean J. Chem. Eng.*, 10, pp. 15–29.
- [16] Seo, K., Hu, J., Fan, Z., Goodman, E. D., and Rosenberg, R. C., 2002, "Automated Design Approaches for Multi-Domain Dynamic Systems Using Bond Graphs and Genetic Programming," *The International Journal of Computers, Systems and Signals*, 13(1), pp. 55–70.
- [17] 2005, ADAMS, MSC Inc. (<http://www.mscsoftware.com/products/adams.cfm?Q=396&Z=397>).
- [18] 2006, "DYMOLA," Dynasim Inc., Lund, Sweden, (<http://www.dynasim.se>).
- [19] Paredis, C. J. J., Calderon, A. D., Sinha, R., and Khosla, P. K., 2001, "Composable Models for Simulation-Based Design," *Eng. Comput.*, 17, pp. 112–128.
- [20] Mathworks Inc., 2005, MATLAB/SIMULINK (<http://www.mathworks.com>).
- [21] McPhee, J. J., 1996, "On the Use of Linear Graph Theory in Multibody System Dynamics," *Nonlinear Dyn.*, 9, pp. 73–90.
- [22] Granda, J., 2001, "Computer Generated Block Diagrams From Bond Graph Models CAMP-G As a Tool Box for SIMULINK," *Proceedings of the International Conference on Bond Graph Modeling and Simulation ICBGM'2001*, Phoenix, AZ.
- [23] Mukherjee, A., and Samantaray, A. K., 2000, "System Modeling Through Bond Graph Object on SYMBOLS 2000," *International Conference on Bond Graph Modeling and Simulation*, Phoenix, AZ, Paper No. ICBGM-01.
- [24] Gawthrop, P. J., and Smith, L. P. S., 1996, *Metamodelling: Bond Graphs and Dynamic Systems*, Prentice-Hall, Englewood Cliffs, NJ.
- [25] Easwar, K., Rouyer, F., and Menon, N., 2002, "Speeding to a Stop: The Finite-Time of a Spinning Disk," *Phys. Rev. E*, 66, 045102(R).
- [26] Sendur, P., Stein, J. L., Louca, L. S., and Peng, H., 2003, "An Algorithm for the Assessment of Reduced Dynamic System Models for Design," *Proceedings of the International Conference on Simulation and Multimedia in Engineering Education*, Orlando, FL, Society for Modeling and Simulation International Simulation, San Diego, CA, pp. 92–101.
- [27] Houck, C., Joines, J., and Kay, M., 1995, "A Genetic Algorithm for Function Optimization: A Matlab Implementation," NCSU-IE Technical Report No. 95-09.
- [28] Ulrich, K. T., and Eppinger, S. D., 2004, *Product Design and Development*, 3rd ed., McGraw-Hill, New York.
- [29] Stein, J. L., and Louca, L. S., 1995, "A Component-Based Modeling Approach for System Design: Theory and Implementation," *Proceedings of the 1995 International Conference on Bond Graph Modeling and Simulation*, Las Vegas, NV, SCS, San Diego, CA, Vol. 27, pp. 109–115.
- [30] Wilson, B. H., and Stein, J., 1995, "An Algorithm for Obtaining Minimum Order Models of Distributed and Discrete Systems," *ASME J. Dyn. Syst., Meas., Control*, 41, pp. 47–58.