# Concept generation from the functional basis of design

**Article** · January 2005

CITATIONS
72

READS
3,778

**5 authors**, including:

Daniel A. Mcadams
Texas A&M University
**171** PUBLICATIONS **3,704** CITATIONS
SEE PROFILE

Tolga Kurtoglu
Palo Alto Research Center
**67** PUBLICATIONS **1,767** CITATIONS
SEE PROFILE

Matthew I. Campbell
Oregon State University
**163** PUBLICATIONS **2,318** CITATIONS
SEE PROFILE

INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 05 MELBOURNE, AUGUST 15-18, 2005

# CONCEPT GENERATION FROM THE FUNCTIONAL BASIS OF DESIGN

Cari R. Bryant, Robert B. Stone, Daniel A. McAdams, Tolga Kurtoglu, Matthew I. Campbell

*Keywords: concept generation, functional basis, conceptual design, functional models*

## 1 Introduction

Few computational tools exist to assist designers during the conceptual phase of design, and design success is often heavily weighted on personal experience and innate ability. Many well-known methods (e.g. brainstorming, intrinsic and extrinsic searches, and morphological analysis) are designed to stimulate a designer's creativity, but ultimately still rely heavily on individual bias and experience. Under the premise that quality designs comes from experienced designers, experience in the form of design knowledge is extracted from existing products and stored for reuse in a web-based repository. This paper presents a concept generation algorithm that utilizes the Functional Basis and a web-based repository of existing design knowledge to generate and rank viable conceptual design variants. This tool is intended to augment traditional conceptual design phase activities and produce numerous feasible concepts early in the design process.

## 2 Background

The concept generation phase of the design process is, at best, difficult to translate into a succinct methodology that is useful to both experienced and inexperienced designers. Quantification and formalization of the conceptual design phase is an active, but immature, area of research. Many formal methods of conceptual design have yet to be realized as computational algorithms. The work presented in this paper shows the initial steps taken to generate a matrix-based algorithm for concept generation and early concept evaluation. The specific focus of this research is the combination and formalization of function-based synthesis, constraint management, and state space search to create a comprehensive space of concept variants and search it for feasible design candidates.

Existing design tools primarily focus on the initial design phases, such as customer need gathering (e.g. quality function deployment), or on the later steps of design embodiment or detail design (e.g. design structure matrices, graph grammars, solid models, dynamic modeling, and finite element analysis.) Few computational tools exist to assist designers during the conceptual phase of design. Instead, designers have limited options available for creating a quality design. Available options may include drawing on personal experiences or the experiences of co-workers, utilizing patent searches to find other approaches or similar designs, and reverse engineering existing products to evaluate how either the current design or a redesign could be used to meet the design goals. All of these methods are potentially limited or biased by a designer's experiences. In addition, patent searches and reverse engineering are potentially time intensive and laborious and may not catch solutions that seem unrelated but are, in fact, analogous.

Innovations cited by Antonnsson and Cagan [1] indicate that certain parts of larger design problems can be solved automatically and without human expertise. However, automations in the design process are often only employed once basic design concepts have

been selected but lack specific dimensions. Complete automation of the design process seems to be restricted by a lack of continuity between conceptual design methods and computational design tools. We propose a computationally based method of concept generation that quickly produces a manageable array of concept variants. The automatically generated concept variants can then be used for concept selection or as a catalyst for generating additional concept variants through complimentary non-computational creative techniques. The following sections present a review of the product representation and design tools that have been used in this paper. We then present our computer-implemented algorithm for automatically producing conceptual designs and illustrate its effective use to generate viable concept variants.

# 3 Related Work

We begin with a limited review of the state of the art in area of conceptual design research and areas that support automated concept generation. In particular, we first review systematic approaches to conceptual design and then focus on product function representation and design knowledge collection.

The fuzzy front end of the conceptual design process has seen few attempts at automation, perhaps due to the evolving strategies and methodologies that exist for this phase of design. However, over the past few decades, design methods have matured and systematic approaches to conceptual design have. These design methods provide a starting point for automating the conceptual design phase. In particular, the systematic approach of Pahl and Beitz and Hubka [2, 3], representing European schools of design, has spawned variant methodologies in American design literature. Regardless of the methodology variation, all begin by formulating the overall product function and breaking it into small, easily solved sub-functions. Solutions to the sub-functions are sought and the form of the device then follows from the assembly of all sub-function solutions.

The lack of a precise definition for *small, easily solved sub-functions* has spurred research into the development of a high level design language (sometimes called a vocabulary or taxonomy) to describe product function and thus enable a systematic approach to functional modeling. Hundal [4] formulates six function classes complete with more specific functions in each class, though he does not claim to have an exhaustive list of mechanical design functions. Another approach uses the 20 subsystem representations from living systems theory to represent mechanical design functions [5]. Malmqvist, *et al*. [6] compare the Soviet Union era design methodology known as the Theory of Inventive Problem Solving (TIPS) with the Pahl and Beitz methodology. TIPS uses a set of 30 functional descriptions to describe all mechanical design functions. Malmqvist, *et al*. note that the detailed vocabulary of TIPS would benefit from a more carefully structured class hierarchy using the Pahl and Beitz functions at the highest level. Kirschman and Fadel [7] propose four basic mechanical functions groups, but vary from the standard verb-object sub-function description common to most methodologies. This work appears to be the first attempt at creating a common vocabulary of design that leads to common functional models of products. Japanese researchers have also explored a consistent language for describing the functionality of products and relating it to product behavior.

Recent work continues this pursuit for a standard language that subsumes the previous work. The result of these recent efforts is a design language known as the Functional Basis [8]. The Functional Basis uses the function and flow words in Tables 1 & 2 to form a sub-function description as a *function* and a *flow* (i.e., a verb-object format). The Functional Basis is intended to be broad enough to span the entire mechanical design space while not being repetitive. In Table 1, engineering *flows* are categorized as three classes (material, signal and

energy) and then further specified as basic categories within each class. In Table 2, engineering *functions* are categorized as 8 classes that are further specified as basic categories.

***Table 1:*** *Flow classes and their basic categorizations.*

| Class | Material | Signal | Energy | | |
|---|---|---|---|---|---|
| ***Secondary*** | Human | Status | Human | Electrical | Mechanical |
| | Gas | Control | Acoustic | Electromagnet | Pneumatic |
| | Liquid | | Biological | Hydraulic | Radioactive |
| | Solid | | Chemical | Magnetic | Thermal |
| | Plasma | | | | |
| | Mixture | | | | |

***Table 2:*** *Function classes and their basic categorizations.*

| Class | Branch | Channel | Connect | Control Magnitude | Convert | Provision | Signal | Support |
|---|---|---|---|---|---|---|---|---|
| ***Secondary*** | Separate | Import | Couple | Actuate | Convert | Store | Sense | Stabilize |
| | Distribute | Export | Mix | Regulate | | Supply | Indicate | Secure |
| | | Transfer | | Change | | | Process | Position |
| | | Guide | | Stop | | | | |

A more formal approach to functional representation stores customer needs and functions in a matrix format [9, 10]. This process is shown schematically in Figure 1. In this work, products are represented quantitatively as vectors that indicate the functionality present in each product. Multiple products are aggregated together to form a product-function matrix, $\Phi$. Matrix manipulations of $\Phi$ produce useful design knowledge such as similarity of products based on functionality and customer needs (using the formulation $\Lambda=\Phi^T\Phi$). This computational approach provides a way to 'design by analogy,' i.e., use stored product knowledge to design new or redesign existing products.

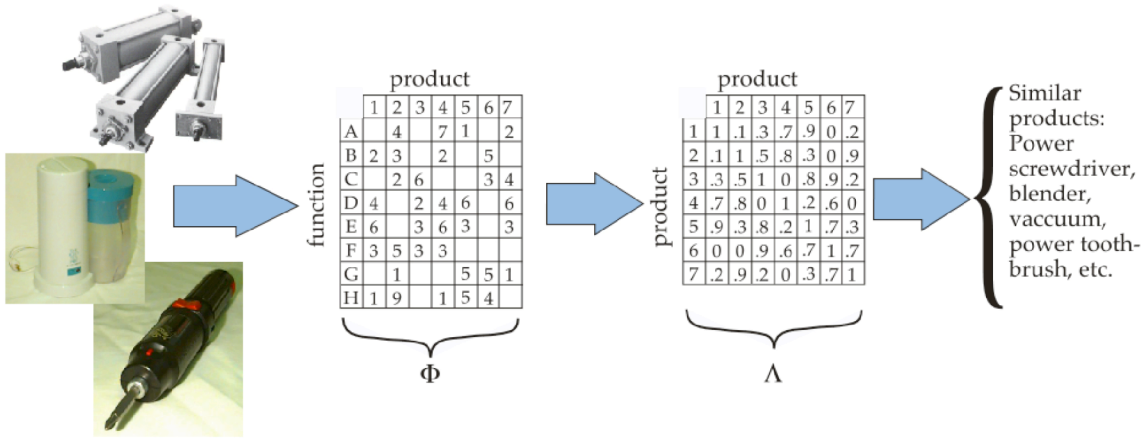



***Figure 1:*** *A schematic representation of creating a product-function repository and its manipulation to produce a product-product matrix and the corresponding product families.*

Building on this representation, a prototypical design repository following the NIST-proposed format has been developed [11, 12, 13]. The repository, in its current form, holds information on approximately 50 consumer products and, following the NIST schema, identifies artifact, function, flow, form, geometry, material, behavior and specification for each product in an XML format [13].

From a perspective different than the functional modeling approach discussed above, a number of research efforts have sought to establish a generic computational scheme for electromechanical design. While these methods have yet to capture *function* on the same level understood by human designers, such approaches have been used in attempts to synthesize new electromechanical configurations. In approaches found in literature, the repeating refrain is that computational synthesis approaches produce an overwhelming number of concept variants – even for limited domains – and a pruning method is needed to realistically identify

all feasible solutions. The automated concept generation program illustrated in Section 4 utilizes strategies comparable to the above matrix techniques [9, 10] to reuse existing stored design knowledge to quickly produce and evaluate many feasible concepts.

# 4 Review of Design Tools Used During Concept Generation

The following sections describe the design tools used to automate the concept generation phase of the design process. First, the Functional Basis is presented as the means of both capturing the design knowledge and representing the conceptual design input required by the computational scheme. Next, the web-based design repository used to store the design knowledge is described. Lastly, the matrix manipulations that comprise the concept generation algorithm are described.

## 4.1 Functional Basis of Design

Intended to span the entire mechanical design space without repetition, the Functional Basis uses function and flow words to form a sub-function description as a *function* and a *flow* (i.e., a verb-object format). Generation of a black box model, creation of function chains for each input flow, and aggregation of function chains into a functional model are the sequence of steps that lead to the repeatable formation of a functional model in their approach. To briefly illustrate this technique, the functional model of an insulating cup is shown in Figure 2. The black box model is constructed based on the overall product function and includes the various energy, material, and signal flows involved in the global functioning of the product. The detailed functional model is then derived from sub-functions that operate on the flows listed in the black box model.

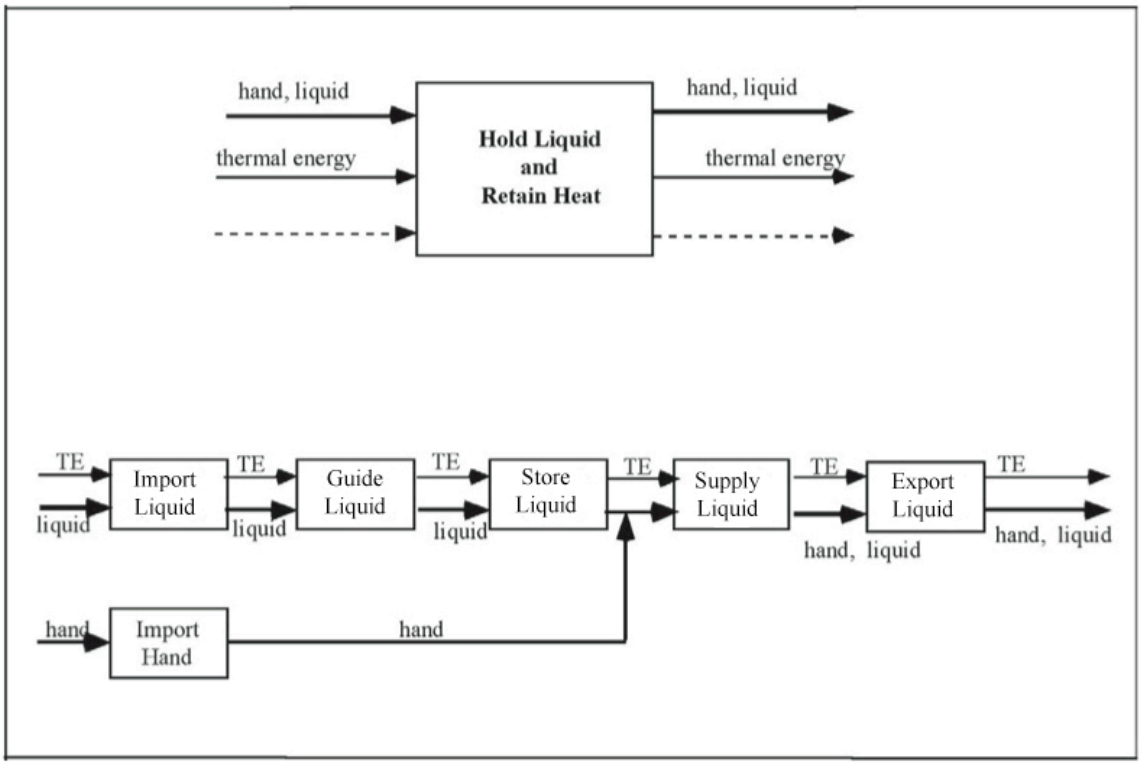



***Figure 2:*** *Black box model and functional model of a cup.*

Functional models for any product can be generated using this technique. Repeatability, ease in storing and sharing design information, and increased scope in the

search for solutions are some of the advantages of these functional. Functional models reveal functional and flow dependencies and are used to capture design knowledge from existing products for inclusion in the web-based design repository [13]. Functional representations also increase the clarity of the design problem and tracking of input and output flows [2].

## 4.2 Design Repository

Over the course of several years, a web-based repository to store design knowledge has been developed and refined at the University of Missouri–Rolla and in collaboration with the University of Texas at Austin [14]. This repository, which includes descriptive product information such as functionality, bills of materials, and design structure matrices (DSMs), now contains detailed design knowledge on approximately 50 consumer products. The knowledge contained in the repository is steadily expanding and benefits from a broad base of consumer products. Design tools like function component matrices (FCMs) and design structure matrices (DSMs) can be readily generated from single or multiple products and used in a variety of ways to enhance the design process [13, 14]. Next, we discuss more specifically how these design repository tools can be utilized to produce viable concept variants.

# 5 Concept Generation Algorithm

Our proposed method utilizes the Functional Basis to link component functionality with component compatibility and create, filter, and rank concept variants [8, 15]. Generated from a web-based repository of design information, the function-component matrix (FCM) and the design structure matrix (DSM) describe the function-component relationships and the component-component compatibility of existing consumer products [15, 16]. Product descriptions stored in the database allow access to information such as historical occurrence and failure mode, which help limit and rank design solutions. The following section details the algorithm that uses the design knowledge contained in the repository to generate, filter, and rank concept variants for further analysis by design engineers. In addition, Figure 3 graphically summarizes the theory behind each step in the concept generation scheme and relates it to the matrix-based tools that can be produced and manipulated to compute the set of filtered concept variants.

## 5.1 Step 1: Generate a Conceptual Functional Model

The proposed concept generation scheme begins with the functional model for either a new product to be developed or a previously developed product that is to be redesigned. Using the Functional Basis technique presented in Section 4.1, a graphical block diagram that defines the flows through the product and the functions that act on those flows is created. This block diagram is the then translated into a matrix form that describes the connectivity between functions. Step 1 under Theory in Figure 3 shows a simple generic flow chain of the form used to create functional models using the Functional Basis method, where f1-f4 are unspecified sub-functions of the product to be designed. Figure 3 also illustrates the matrix equivalent of this flow chain, which is a connectivity matrix where a non-zero cell entry indicates a forward connection between the row and column functions.

## 5.2 Step 2: Define Function-Component Relationships Using Existing Design Knowledge

The next step utilizes design knowledge gathered from existing consumer products to define the relationships between a component and the functions that it solves in the examined products. Reverse engineering techniques are applied to existing consumer products, and information extracted from each product's bill of materials and functional model is stored in the web-based design repository described in Section 4.2. Information describing the functionality of each artifact is stored in the online database, and function-component matrices for individual products or specified groups of products can easily be generated from the stored information. Non-zero cell entries in the function-component matrix indicate that component from the column containing the cell can solve the function in the row containing the cell. Step 2 in Figure 3 shows how the function-component matrix equivalent describes the function-component relationships in the sample shown under the Theory column.

## 5.3 Step 3: Compute the Set of Conceptual Variants that Solve the Function Model

Step 3 utilizes the information from Step 1 and Step 2 to create a set of design solutions. In step 3 under Theory in Figure 3, a component "tree" is created showing the chains of components that could potentially solve the flow chain presented in Step 1, based on the component-function relationship information shown in step 2. Although the example illustrated in Figure 3 results in a single branching tree for step 2, it is important to note that multiple branching trees may be formed at this stage when multiple components have the potential to solve the initiating function in the chain. Computationally, if the transpose of the row vector from the function-component matrix (FCM) that corresponds to each of the functions from the flow chain in Step 1 is matrix multiplied by the row vector from the FCM that corresponds to the forward connected function, a component-component matrix will be generated for each function connection in the flow chain. This matrix multiplication is illustrated as the matrix equivalent to Step 3 in Figure 3. Non-zero cells within these newly created component-component matrices represent all theoretically possible component combinations that will solve each pair of functions in the flow chain. If these component-component matrices are then placed into the connectivity matrix, component paths can be traced through the aggregated matrix much the same way a path is traced along the tree shown in Step 3 under Theory in Figure 3. Tracing every possible "path" of connections will give a list of all theoretically possible component chain variations that solve the function chain presented in Step 1 of Figure 3.

## 5.4 Step 4: Define Component-Component Compatibility Using Existing Design Knowledge

The next step uses additional design knowledge gathered from existing consumer products to define the compatibility between components in the examined products. As each product is reverse engineered, information regarding the connection between components is extracted from assembly models and stored in the web-based design repository described in Section 4.2. Connection data for each artifact is stored in the online database, and component-component matrices for individual products or specified groups of products can easily be generated from the stored information. Non-zero cell entries in the component-component matrix (frequently called a design-structure matrix or DSM) indicate that the component from

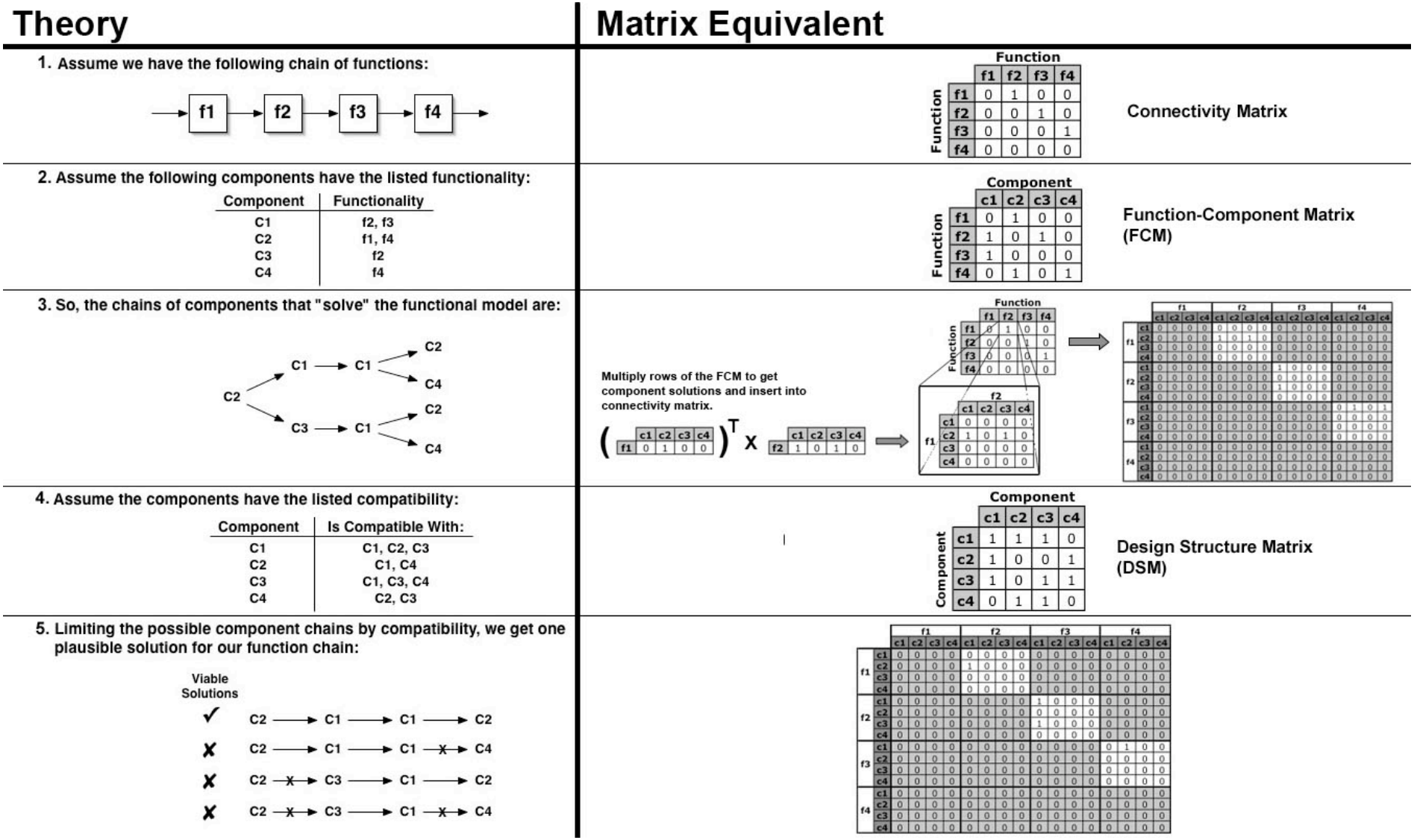



***Figure 3:*** *Visual summary of the concept generation algorithm.*

the column containing the cell has been directly connected to the component in the row containing the cell in an existing product. Step 4 in Figure 3 shows how the design-structure matrix equivalent describes the component-component connectivity in the sample shown under the Theory column.

### 5.5 Step 5: Filter Set of Conceptual Variants Using Component Compatibility Knowledge

Step 5 uses the component compatibility information contained in the design-structure matrix (DSM) to prune the tree of design solutions computed in Step 3. Shown under the Theory column for Step 5 in Figure 3, each component connection in each component chain is checked for compatibility using the existing connection information from Step 4. An 'X' has been overlaid on each component connection line that is not supported by the data shown in the previous step. In the matrix equivalent, each cell of the DSM is multiplied with the corresponding cell in each of the function pair component-component matrices generated in Step 3. Overlaying the DSM on each matrix created in step 3 (via cell multiplication) has the effect of removing any of the possible component connections that do not appear in the repository database. This technique uses the "experience" contained in the repository to filter out potentially inadequate concept variants and reduce the set of possible concept variants down to a more manageable size. After the matrices are filtered, we can once again trace every "path" of possible components to generate a list of feasible component chains that solve the function chain from Step 1.

Finally, this final list of feasible solutions can be ranked to bubble the most promising solutions to the top of the list based on a designer's specified needs. For instance, various measures of design needs (e.g. manufacturability, recyclability, failure etc.) entered as the non-zero FCM and/or DSM entries can be used to rank and sort the resulting conceptual design solutions generated by this method. Once the set of filtered concept variants has been computed and ranked, a designer is then free to sift through the generated concept variants and evaluate the application of each to the design situation at hand.

The algorithm illustrates a method to quickly produce and sort a set of conceptual designs for a new or redesigned product. Functions comprising a proposed product's functional model are mapped to lists of components that are capable of solving each function. The tree of possible component chains is then pruned by eliminating infeasible component connections according to component-component compatibility. This filtered set of component chains is then ranked and presented to the design engineer for further analysis. The following section takes the presented algorithm and manually applies it to a simplified design example using Tinkertoy parts as the set of components available in a simulated design repository.

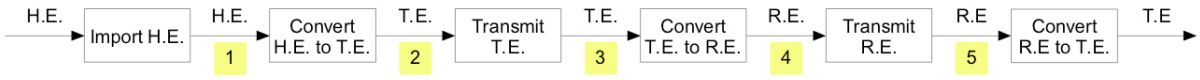
## 6 Illustrative Example

A tricycle built from a standard Tinkertoy set, shown in Figure 4, is next presented as a simple proof of-concept example. This simplified example demonstrates the effectiveness of the described methodology while utilizing a manageable set of artifacts for ease of illustration.

***Figure 4:*** *Tinkertoy tricycle used as the "product" to be redesigned in the following example.*

First, a functional model of the tricycle construction was generated as described in Step 1 of the concept generation algorithm. For demonstration purposes, the subsequent steps of the concept generation scheme were only applied to the energy flow chain, shown in Figure 5a, from the complete functional model. The functional model of this flow chain begins by importing human energy across the product boundary of the tricycle toy. The model follows the energy flow as it gets converted to translational energy and transmitted through the product, then gets converted into rotational energy, which is further transmitted through the product and finally converted back into translational energy. Figure 5b shows the function connectivity matrix generated from the energy function chain in Figure 5a. Sequential numbers were used in the function connectivity matrix for easy reference to the connections labeled in the energy function chain.

**(a)**




*Key:* *H.E. ≡ Human Energy*
*T.E. ≡ Translational Energy*
*R.E. ≡ Rotational Energy*

**(b)**

| | Import H.E. | Transmit R.E. | Transmit T.E. | Convert H.E. to | Convert H.E. to T.E. | Convert R.E. to T.E. | Convert T.E. to R.E. |
|---|---|---|---|---|---|---|---|
| **Import H.E.** | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| **Transmit R.E.** | 0 | 0 | 0 | 0 | 0 | 5 | 0 |
| **Transmit T.E.** | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| **Convert H.E. to R.E.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Convert H.E. to T.E.** | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| **Convert R.E. to T.E.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Convert T.E. to R.E.** | 0 | 4 | 0 | 0 | 0 | 0 | 0 |

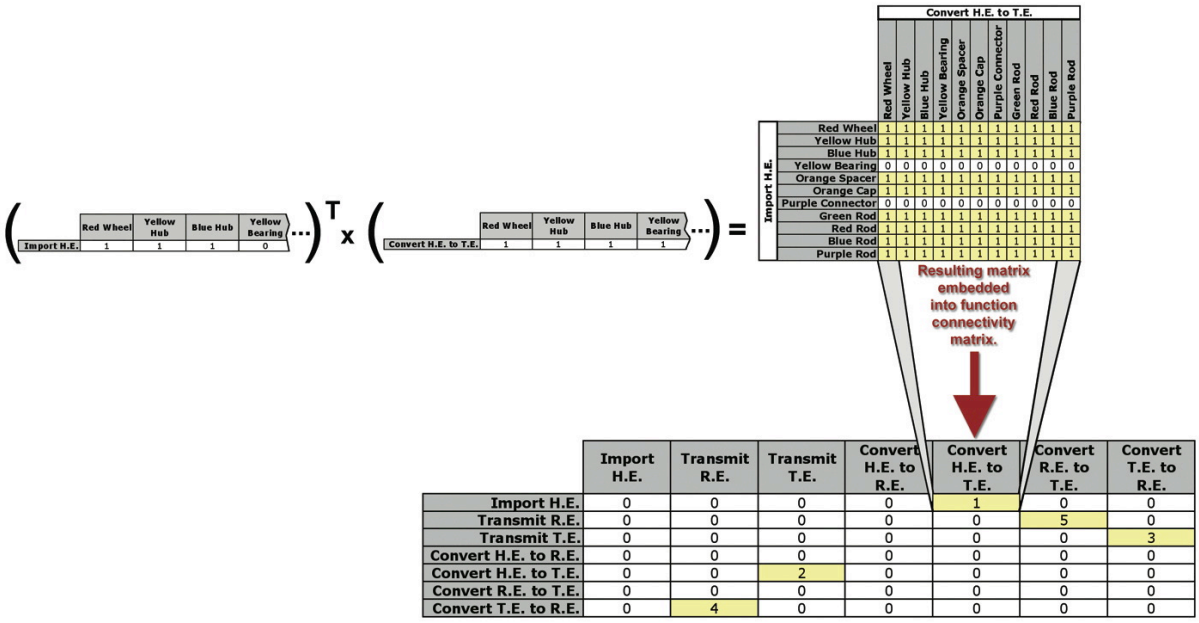
***Figure 5:*** *(a) Function chain for the energy flow through the Tinkertoy tricycle.* ***(b)*** *Function connectivity matrix that describes the function connections graphically shown in (a).*

Next, in Step 2 of the concept generation scheme, a function-component matrix (FCM) was constructed for the complete set of Tinkertoy components. In contrast to the web-based repository method of generating a FCM from existing products, the FCM was generated by subjectively defining the functionality by inspection of the Tinkertoy component set, which is, in effect, a mini-repository of Tinkertoys. First, a list of all relevant function/flow combinations was selected from a FCM generated for all product design knowledge that is currently included in the web-based repository. Then, each component in the Tinkertoy set was evaluated for its potential to solve each function from the subset list. The complete FCM generated for the Tinkertoy set is shown below in Figure 6. For instance, we can see that the yellow bearing component is capable of embodying the following functionality: Guiding a solid, distributing translational energy, transmitting translational energy, converting human energy to translational energy, and converting translational energy to rotational energy.

| | Red Wheel | Yellow Hub | Blue Hub | Yellow Bearing | Orange Spacer | Orange Cap | Purple Connector | Green Rod | Red Rod | Blue Rod | Purple Rod |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Import Solid** | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| **Translate Solid** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Guide Solid** | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Rotate Solid** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Support Solid** | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| **Stabilize Solid** | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| **Secure Solid** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Position Solid** | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Distribute T.E.** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Import H.E.** | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| **Transmit R.E.** | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Transmit T.E.** | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Convert H.E. to R.E.** | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **Convert H.E. to T.E.** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Convert R.E. to T.E.** | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **Convert T.E. to R.E.** | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

***Figure 6:*** *Function-component matrix generated for the set of Tinkertoy components.*

Using the function connectivity information from Figure 5b and the component functionality from Figure 6, the entire set of theoretical concept variants for the redesign was calculated for Step 3 of the concept generation algorithm. As illustrated in Figure 7, rows for each of the connected function pairs were multiplied together to generate the unfiltered matrices of design solutions for each function pair. These unfiltered matrices are then embedded into the function connectivity matrix to describe the full set of theoretical solutions.




| | Import H.E. | Transmit R.E. | Transmit T.E. | Convert H.E. to R.E. | Convert H.E. to T.E. | Convert R.E. to T.E. | Convert T.E. to R.E. |
|---|---|---|---|---|---|---|---|
| Import H.E. | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Transmit R.E. | 0 | 0 | 0 | 0 | 0 | 5 | 0 |
| Transmit T.E. | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| Convert H.E. to R.E. | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Convert H.E. to T.E. | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| Convert R.E. to T.E. | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Convert T.E. to R.E. | 0 | 4 | 0 | 0 | 0 | 0 | 0 |

***Figure 7:*** *Matrix row multiplication is used to generate the set of theoretical design solutions for each connected function pair. Resulting matrices are embedded in the function connectivity matrix.*

In Step 4, a similar method to that used to create the FCM was employed to construct the design structure matrix (DSM) for the set of Tinkertoy components. The DSM, shown in Figure 8, describes the component compatibility between each component, where 1's entered into each cell identifies components that can be connected together, and 0's indicate incompatibility.

| | Red Wheel | Yellow Hub | Blue Hub | Yellow Bearing | Orange Spacer | Orange Cap | Purple Connector | Green Rod | Red Rod | Blue Rod | Purple Rod |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Red Wheel** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Yellow Hub** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Blue Hub** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Yellow Bearings** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Orange Spacers** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Orange Caps** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **Purple Connectors** | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **Green Rods** | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **Red Rods** | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **Blue Rods** | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| **Purple Rods** | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

***Figure 8:*** *Design structure matrix (DSM) generated for the set of Tinkertoy components.*

Finally, in Step 5, each cell of the DSM was multiplied by the corresponding cell for each of the connected function pairs in order to filter out design solutions that are infeasible due to component incompatibility. The entire set of filtered design solutions is shown in Figure 9. To clarify the pertinent information, cells that contained zero values in the original function connectivity matrix are grayed out. Additionally, non-zero entries that indicate feasible component combinations for each connected function pair are highlighted in yellow.

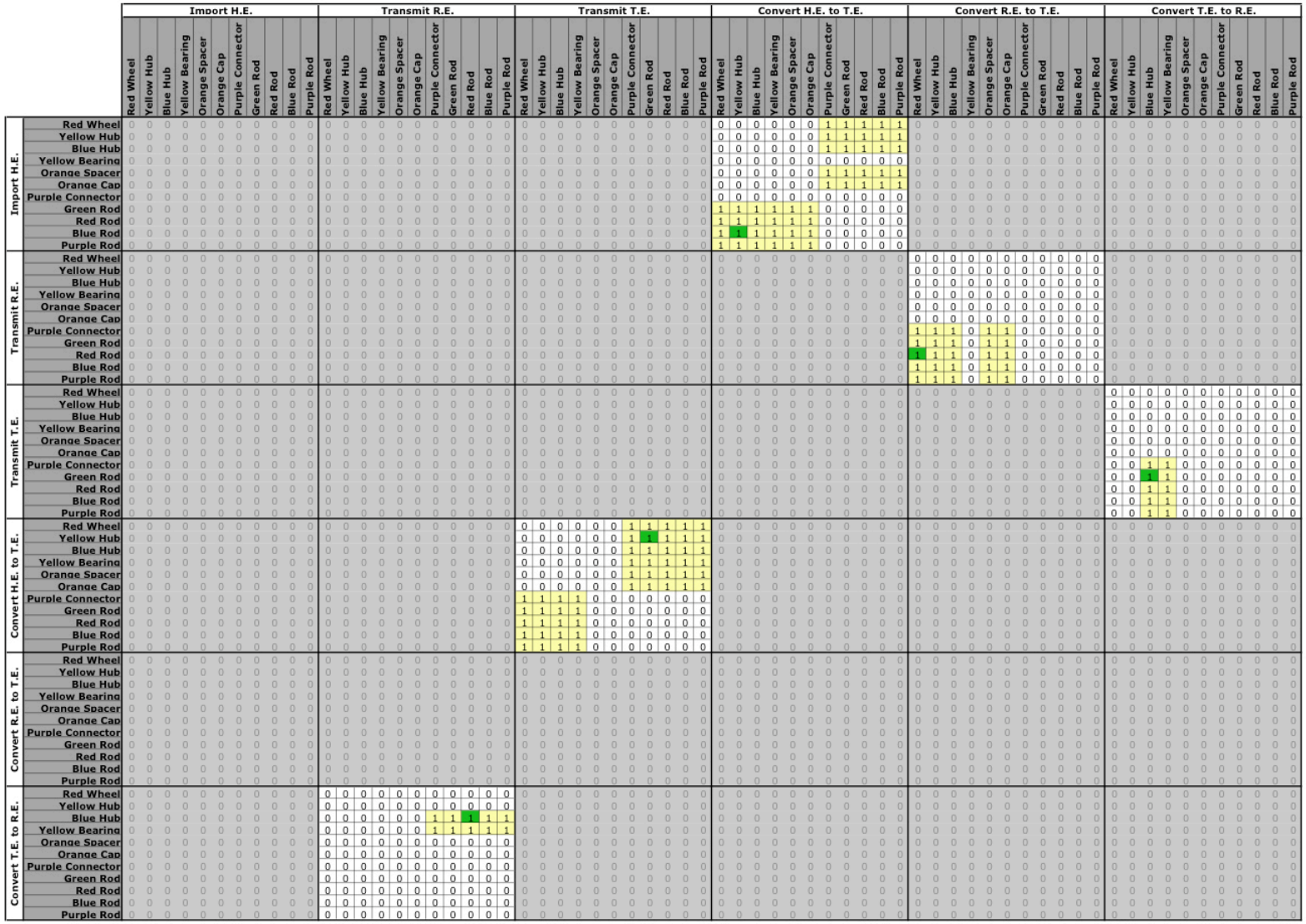



***Figure 9:*** *Function connectivity matrix with embedded component connection information that describes the complete set of feasible design solutions for the tricycle redesign.*

Figure 10a-10d presents four of the design variants encompassed in the matrix presented in Figure 9. The design variants shown are unstable asymmetric versions of the original tricycle concept since the energy function chain generated in Step 1 does not encompass requirements that the design be stable. The design variant in Figure 10a was constructed by selecting the component connections highlighted in green in Figure 9. For instance, using Figure 9, we can look at the embedded matrix describing potential component connections that simultaneously solve the functions import human (i.e. the row defined as Import H.E.) and convert human energy to translational energy (i.e. the column defined as Convert H.E. to T.E). From this matrix, the green highlighted entry indicates that a blue rod connected to a yellow hub will successfully embody these two connected functions. Next, examining the embedded matrix contained by the row defined as convert human energy to translational energy (Convert H.E. to T.E.) and the column defined as transmit translational energy (Transmit T.E.), we can next choose a component compatible with the already selected yellow hub (e.g. the green rod cell highlighted in green) to solve the next pair of connection function pairs in the chains. Continuing in a similar fashion produces a chain of components that solves the function chain generated in Step 1.

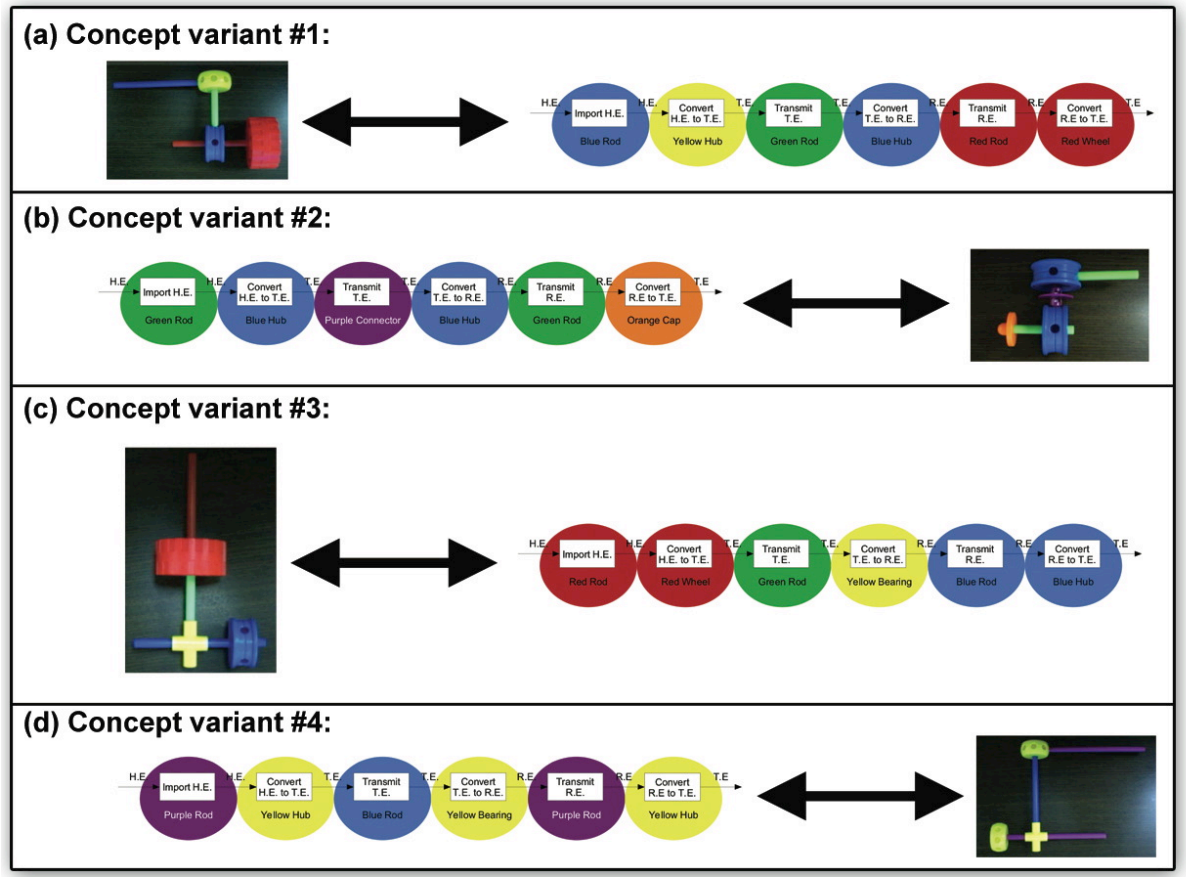



***Figure 10: (a)-(d)*** *Concept variants selected from the matrix of feasible solutions presented in Figure 9.*

Using this technique, we can identify all possible feasible design solutions for the product to be designed or redesigned. Although not feasible due to the manual generation of the FCM and DSM for the simple example presented here, ranking of the design solutions can be accomplished by calculating a "score" for each concept variant based on the component connections selected using stored measures of frequency of occurrence, manufacturability, assemblability, etc.

# 7 Conclusions

Application of our concept generation algorithm to the tricycle design example has demonstrated the capability to automatically produce viable conceptual design variations based on existing component knowledge. Although not presented here, application of the presented methodology to actual product design and redesign results in similar success in producing concept variants. Future work will focus on fine tuning the process and generating conceptual variants for multiple interacting functional chains. In addition, the algorithm will incorporate identification of potential function sharing opportunities looking for instances where connected functions can be solved by the same component. Further improvements to the algorithm can address product functionality that is typically solved at the assembly level rather than the component level of a product, for instance balance and stabilization. Next, issues with the number of potential connections a component can handle and loss of component functionality following connection to another component can be addressed through the use of port information [17] and assembly diagrams. Finally, other areas of expansion include compensation for novel solutions that might stem from component compatibility and functionality that may be possible although not historically found and inclusion of manufacturing needs in addition to customer needs to seed the concept generation of a product.

**References**


[1] Antonsson, E. K., Cagan, J., 2001, Formal Engineering Design Synthesis, Cambridge University Press.

[2] Pahl, G. and Beitz, W., 1988, Engineering Design: A Systematic Approach, Springer-Verlag.

[3] Hubka, V. and Ernst Eder, W., 1984, Theory of Technical Systems, Springer-Verlag, Berlin.

[4] Hundal, M., 1990, "A Systematic Method for Developing Function Structures, Solutions and Concept Variants," Mechanism and Machine Theory, 25(3):243-256.

[5] Koch, P., Peplinski, J., Allen, J. and Mistree, F., 1994, "A Method for Design Using Available Assets: Identifying a Feasible System Configuration," Behavioral Science, 30:229-250.

[6] Malmqvist, J., Axelsson, R., and Johansson, M., 1996, "A Comparative Analysis of the Theory of Inventive Problem Solving and the Systematic Approach of Pahl and Beitz," Proceedings of the 1996 ASME Design Engineering Technical Conferences, 96-DETC/DTM-1529, Irvine, CA.

[7] Kirschman, C. and Fadel, G., 1998, "Classifying Functions for Mechanical Design," Journal of Mechanical Design, Transactions of the ASME, 120(3):475-482.

[8] Hirtz, J., Stone, R., McAdams, D., Szykman, S. and Wood, K., 2002, "A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts," Research in Engineering Design, 13(2):65-82.

[9] McAdams, D., Stone, R., and Wood, K., 1999, "Functional Interdependance and Product Similarity based on Customer Needs," Research in Engineering Design, 11(1)1-19.

[10] Stone, R., Wood, K., and Crawford, R., 2000, "Using Quantitative Functional Models to Develop Product Architectures," Design Studies 21(3):239-260.

[11] Szykman, S., 2002, "Architecture and Implementation of a Design Repository System," Proceedings of DETC2002, DETC2002/CIE-34463, Montreal, Canada.

[12] NIST Workshop on Product Representation for Next-Generation Distributed Product Development, National Institute of Standards and Technology, Gaithersburg, MD, Nov. 30 - Dec. 1, 2000.

[13] Bohm, M. and Stone, R., 2003, "Refining Design Repositories: Creating a Usable Framework with XML Data Representation," Proceedings of the 2003 NSF Grantees Conference, Birmingham, AL.

[14] Bohm, M., and Stone, R., 2004, "Representing Functionality to Support Reuse: Conceptual and Supporting Functions," Proceedings of DETC'04, DETC2004-57693, Salt Lake City, UT.

[15] Strawbridge, B., McAdams, D. and Stone, R., 2002, "A Computational Approach To Conceptual Design," Proceedings of DETC2002, DETC2002/DTM-34001, Montreal, Canada.

[16] Pimmler, T. and Eppinger, S., 1994, "Integration Analysis of Product Decompositions," Proceedings of the ASME Design Theory and Methodology Conference, DE-Vol. 68.

[17] Campbell, M.I., Cagan, J., Kotovsky, K. 1999. "A-Design: An agent-based approach to conceptual design in a dynamic environment," Research in Engineering Design. 11(3): 172-92.

Robert B. Stone
University of Missouri – Rolla
Basic Engineering Department
107A Basic Engineering
1870 Miner Circle
Rolla, MO, 65409
USA
Phone: (573) 341-4086
Fax: (573) 341-6593
E-mail: rstone@umr.edu