Engineering Optimization

ISSN: 0305-215X (Print) 1029-0273 (Online) Journal homepage: http://www.tandfonline.com/loi/geno20

A novel evolutionary engineering design approach for mixed-domain systems

Zhun Fan , Kisung Seo , Jianjun Hu , Erik D. Goodman & Ronald C. Rosenberg

To cite this article: Zhun Fan , Kisung Seo , Jianjun Hu , Erik D. Goodman & Ronald C. Rosenberg (2004) A novel evolutionary engineering design approach for mixed-domain systems , Engineering Optimization, 36:2, 127-147, DOI: 10.1080/03052150410001647957

To link to this article: http://dx.doi.org/10.1080/03052150410001647957

Published online: 12 May 2010.

Engineering Optimization
Vol. 36, No. 2, April 2004, 127–147

Taylor & Francis
Taylor & Francis Group

A NOVEL EVOLUTIONARY ENGINEERING DESIGN APPROACH FOR MIXED-DOMAIN SYSTEMS

ZHUN FAN[a,*], KISUNG SEO[a], JIANJUN HU[a], ERIK D. GOODMAN[a] and RONALD C. ROSENBERG[b]

[a]*Genetic Algorithms Research and Applications Group (GARAGe);* [b]*Department of Mechanical Engineering, Michigan State University, East Lansing, MI 48824, USA*

This paper presents an approach to engineering design of mixed-domain dynamic systems. The approach aims at system-level design and has two key features: first, it generates engineering designs that satisfy predefined specifications in an automatic manner; second, it can design systems belonging to different or mixed physical domains, such as electrical, mechanical, hydraulic, pneumatic, thermal systems and/or a mixture of them. Two important tools are used in this approach, namely, bond graphs and genetic programming. Bond graphs are useful because they are domain independent, amenable to free structural composition, and are efficient for classification and analysis, allowing rapid determination of various types of acceptability or feasibility of candidate designs. Genetic programming, on the other hand, is a powerful tool for open-ended topological search. To prevent the premature convergence often encountered in evolutionary computation, a hierarchical fair competition model is adopted in this work. Examples of an analog filter design and an MEM filter design illustrate the application of the approach.

Keywords: Mixed-domain design; Bond graphs; Genetic programming; Open-ended topological search

1 INTRODUCTION

Automated generation of system designs to meet given specifications is undoubtedly a very difficult task – the essence of an inverse problem – but there are some very successful examples that demonstrate its feasibility and potential importance. Much research has been done on automated design of single-domain systems using an evolutionary computation approach – for example, automated design of analog circuits. The circuit design examples can be classified into two categories: GA-based and genetic programming (GP)-based. Most GA-based approaches realize topology optimization via a GA and parameter optimization with numerical optimization methods [1]. Some GA approaches also evolve both topology and component parameters; however, they typically allow only a relatively limited number of components to be evolved [2]. Although that work basically achieves good results in analog circuit design, they are not easily extendable to interdisciplinary systems like mechatronic systems.

* Corresponding author. E-mail: fanzhun@egr.msu.edu

Engineering Optimization
ISSN 0305-215X print; ISSN 1029-0273 online
http://www.tandf.co.uk/journals
DOI: 10.1080/03052150410001647957

Genetic programming-based approaches, on the other hand, tend to allow the generation of essentially unbounded topologies [3]. However, they also typically require enormous population sizes and a great deal of computer resources to obtain designs of interesting complexity.

Several challenging issues have to be addressed for automated synthesis of multi-domain systems. First, design of interdisciplinary (multi-domain) engineering systems, such as mechatronic systems, differs from design of single-domain systems, such as electronic circuits, mechanisms, and fluid power systems, in part because of the need to integrate the several distinct domain characteristics in predicting system behavior [4]. Secondly, a mechanism is needed to automatically select useful elements from the building block repertoire, construct them into a system, evaluate the system and then reconfigure the system structure to achieve better performance. This article investigates an approach combining GP and bond graphs to automate the process of design of dynamic systems, especially system-level design, to a significant degree. It is a remarkable fact that models based on apparently diverse branches of engineering science can be expressed using the notation of bond graphs, based on energy and information flow. Using the language of bond graphs, one may construct models of electrical, mechanical, magnetic, hydraulic, pneumatic, thermal, and other systems using only a rather small set of ideal elements as building blocks. As a special form of evolutionary computation, GP is a powerful approach for creating and evolving novel design structures in an open-ended manner. Through definition of a set of constructor functions, a genotype tree is created for each individual in each generation. The process of evaluating the genotype tree maps the genotype into a phenotype *i.e.* to the abstract topological description of the design of a mixed-domain system, using a bond graph along with parameters for each component, if needed. Finally, because there are many considerations in dynamic system design that are not completely captured by a bond graph, physical realization is carried out to relate each abstract element of the bond graph to corresponding components in various physical domains. To improve the topology search capability of GP and to reduce dramatically the amount of computation required to find a set of interesting designs, a special form of parallel GP, the hierarchical fair competition GP (HFC-GP), is used in this article [5].

2 METHODOLOGY

2.1 Bond Graphs

The bond graph is a modeling tool that provides a unified approach to the modeling and analysis of dynamic systems, especially hybrid multi-domain systems including mechanical, electrical, pneumatic, hydraulic, etc. It was developed in the 1960s by Paynter *et al.* [6]. It is the explicit tree-like representation of model topology that makes the bond graph such a good candidate for use in open-ended design search (for example, a 'parallel connection' in an electrical system is represented by a single 0-junction node in a bond graph, and a 'series connection' appears as a single 1-junction node). For details of notation and methods of system analysis related to the bond graph representation, refer to Karnopp *et al.* [7] and Rosenberg [8]. Figure 1 illustrates a bond graph that represents either of the accompanying electrical or mechanical systems. Much recent research has explored the bond graph as a tool for design [9–12]. Design of controllers by augmenting bond graphs with signals (as used in 'block-diagram' representations) has also been widely practiced [13–15].

Bond graphs have four embedded strengths for design applications, namely (1) the wide scope of systems that can be created because of the multi- and inter-domain nature of bond graphs, (2) the efficiency of evaluation of design alternatives, (3) the natural combinatorial features of bond and node components for generation of design alternatives, and (4) ease of

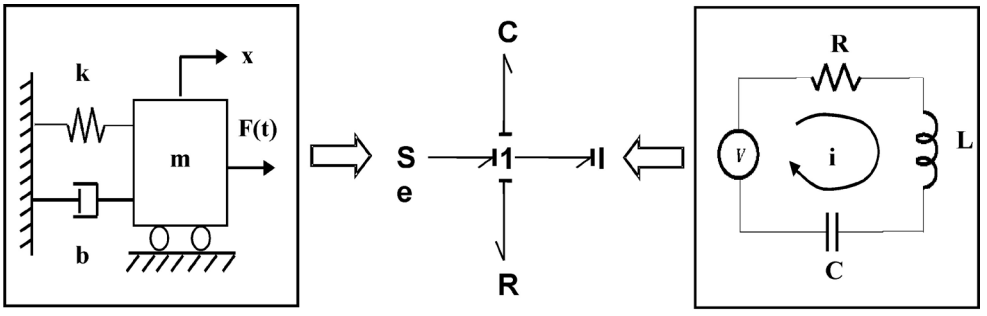

FIGURE 1 Bond graph representation of dynamic systems.

mapping to the engineering design process. Those attributes make bond graphs an excellent candidate for modeling and design of a multi-domain system.

2.2 Bond Graphs and Genetic Programming

Genetic programming is an extension of the genetic algorithm, using evolution to optimize actual computer programs or algorithms to solve some tasks [16, 17], typically involving a graph-type (or other variable-length) representation. The most common form of GP uses trees to represent the entities to be evolved [18]. Genetic programming can manipulate variable-sized strings and can be used to 'grow' trees that specify increasingly complex bond graph models. The tree representation of GP chromosomes, as compared with the string representation typically used in GA, gives GP more flexibility to encode solution representations for many real-world design applications. In the work reported here, the bond graph, which can contain cycles, is not represented directly as the GP tree – instead, the tree (genotype) encodes a constructor for a bond graph (phenotype).

Defining a proper function set is one of the most significant steps in preparing a genetic programming run. It may affect both the search efficiency of genetic programming and the validity of evolved results, and is closely related to the selection of building blocks for the system to be designed. In this research, a basic function set and a modular function set are presented and listed in Tables I and II. Operators in a basic function set are aimed at enabling discovery

TABLE I Operators in a Basic Function Set.

add C	Add a C element to a junction
add I	Add an I element to a junction
add R	Add an R element to a junction
insert J	Insert a 0-junction in a bond
insert J	Insert a 1-junction in a bond
replace	Replace the current element
replace	Replace the current element
replace	Replace the current element
+	Add two ERCs
−	Subtract two ERCs
enda	End terminal for add functions
endi	End terminal for insert
endr	End terminal for replace
erc	Ephemeral random constant

TABLE II Operators in a Modular Function Set.

insert RU	Insert a resonant unit
insert CU	Insert a coupling unit
insert BU	Insert a bridging unit
add RU	Add a resonant unit
insert J01	Insert a 0-1-junction
insert CIR	Insert a special CIR
insert CR	Insert a special CR
Add J	Add a junction compound
+	Add two ERCs
−	Subtract two ERCs
endn	End terminal for add
endb	End terminal for insert
endr	End terminal for replace
erc	Ephemeral Random constant

of primitive building blocks for the system, while operators in a modular function set purport to specify relatively modular and predefined building blocks already incorporating primitive building blocks. Notice that numeric functions are included in both function sets, as they are needed in both cases.

Examples of a basic operator and a modular operator, namely add_R and insert_RU operators, are illustrated in Figures 2 and 3. As illustrated in Figure 2, the R element is added to an existing junction by the add_R function, adding a node with a connecting bond. An R element also requires an additional parameter value ephemeral random constant (ERC). As illustrated in Figure 3, a resonant unit (RU) consisting of I, R, and C components, all attached to a 1-junction, is inserted in an original bond with modifiable site by the insert_RU function. After the insert_RU function is executed, a new RU is created and one additional modifiable site, namely bond (3), appears in the resulting phenotype bond graph, along with the original modifiable site bond

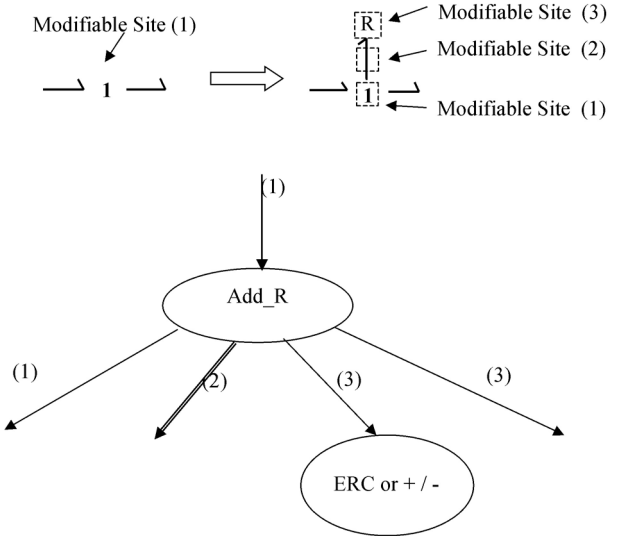

FIGURE 2 Operator to add an R component.

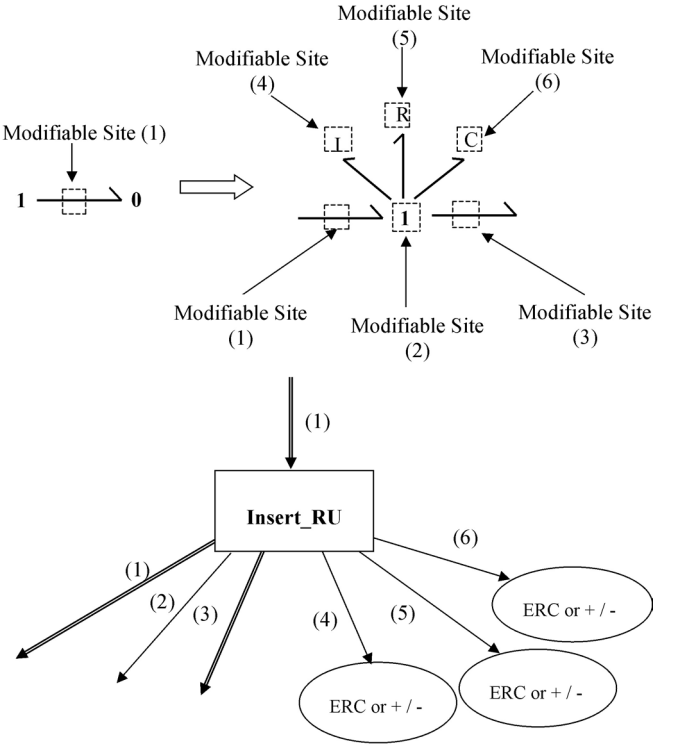

FIGURE 3 Operator to insert a resonant unit.

(1). The new added 1-junction also has an additional modifiable site (2). As the C, I, and R components all have parameters to be evolved, the insert_RU function has three sites (4)–(6) at which numerical parameters will evolve.

Figure 4 shows an example of a GP tree generated at random from the embryo root node. There are three modifiable sites on the embryo, denoted '1' (bond graph node), 'a' (bond), and '2' (bond graph node). Each is specified by an edge of the GP tree. Following edge 1 first shows that an I element (I_3 in Fig. 5) is added by the add_I to the 1-junction (1_1) of the bond graph, together with the I element's parameter value and a new bond. The result is to preserve modifiable site '(1)' and to add modifiable sites '(b)' and '(3)'. The next set of operations under add_I in the GP tree shows that all three sites happen to have been made unmodifiable in the example tree by appending end functions.

Turning next to the edge labeled 'a', it is seen that the first function applied to it is 'end.' That bond site is thereby made unmodifiable. On the other hand, site '(2)' is the locus of additional bond graph growth. A C element, C_4 in Figure 5, is added by add_C to the 0-junction (O_2). In the next operation, insert_J1, a 1-junction (I_5) is inserted between the 0-junction (O_2) and C_4. After the remaining operations, the bond graph of Figure 5 is generated from the GP tree of Figure 4.

2.3 Realizable Function Set

The bond graph/GP approach is a quite general approach to automate synthesizing of multidisciplinary systems. Using a basic set of building blocks allows construction of many types of unconstrained systems. However, engineering systems in the real world are often

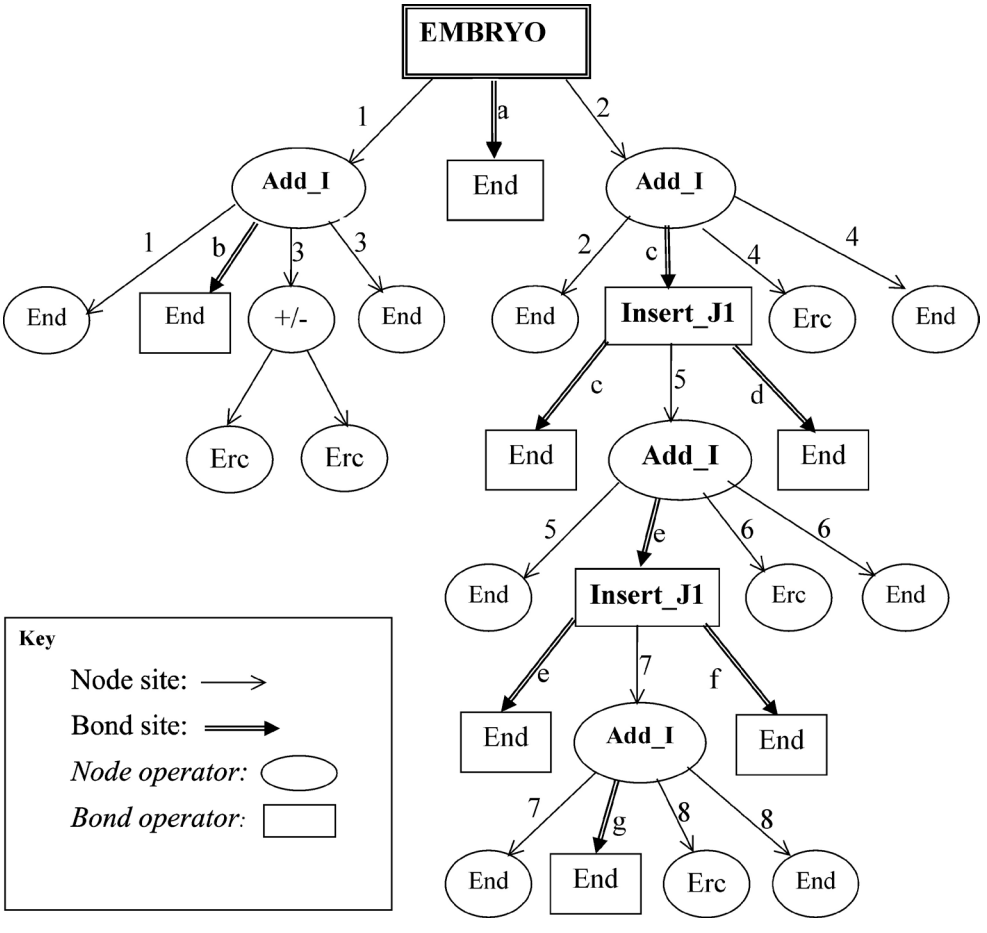

FIGURE 4 Example of a GP tree.

limited by various types of constraints. Synthesizing real-world engineering systems requires satisfying those constraints and integrating their consideration into the approach. The concept of a *realizable function set* is proposed in this paper. Executing only operators in the realizable function set guarantees that the evolved design is physically realizable. This means that not all legal bond graphs will be generated, but that those generated will be physically realizable in the domain of concern. More stringent constraints on manufacturability can also be imposed if needed for a particular application domain.

2.4 Hierarchical Fair Competition Model for Genetic Programming

A special form of parallel GP, hierarchical fair competition (HFC)-GP, is applied in this research. In the HFC model (Fig. 6), multiple subpopulations are organized in a hierarchy, in which each subpopulation can only accommodate individuals within a specified range of fitnesses [5]. New individuals are created continually in the bottom layer. Use of the HFC model balances exploration and exploitation of GP effectively. Experience using the HFC model has shown that it can also substantially increase the topological diversity of the whole population and help to provide the designer with a diverse set of competing design candidates for further trade-offs.

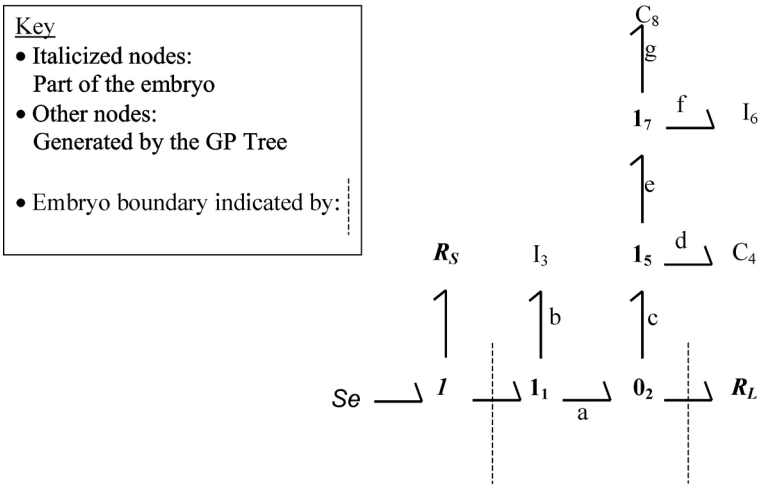

FIGURE 5 Bond graph generated by the example GP tree.

2.5 Discarding Design Candidates Violating Causality Conditions

The design evaluation stage is composed of two steps: (1) causality analysis, and, when merited, (2) dynamic simulation. *Causality* is one of the important concepts in bond graph theory. Causality analysis can give insights into the validity of a model. In causality analysis, the causal relationships and power flow among elements and subsystems can reveal various system properties and inherent characteristics that can make the model unacceptable, and therefore make dynamic simulation unnecessary. While the strong typing used in the GP system will not allow the GP system to formulate 'ill-formed' bond graphs, even 'well-formed' bond graphs can have causal properties that make it undesirable or unnecessary to derive their state models or to simulate the dynamics of the systems they represent. Causality analysis is fast, and can

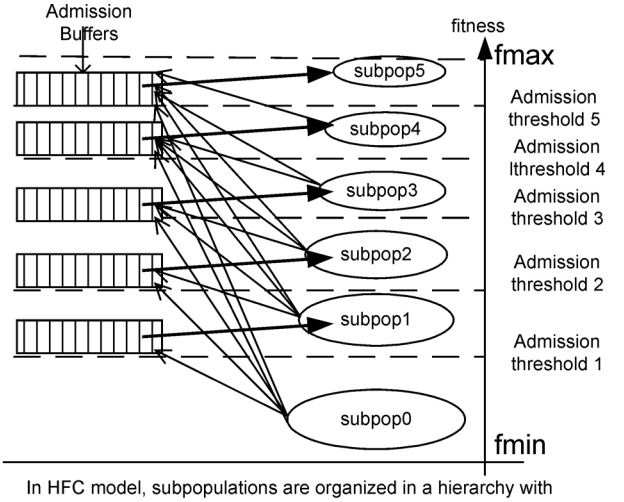

FIGURE 6 Hierarchical fair competition model of GP.

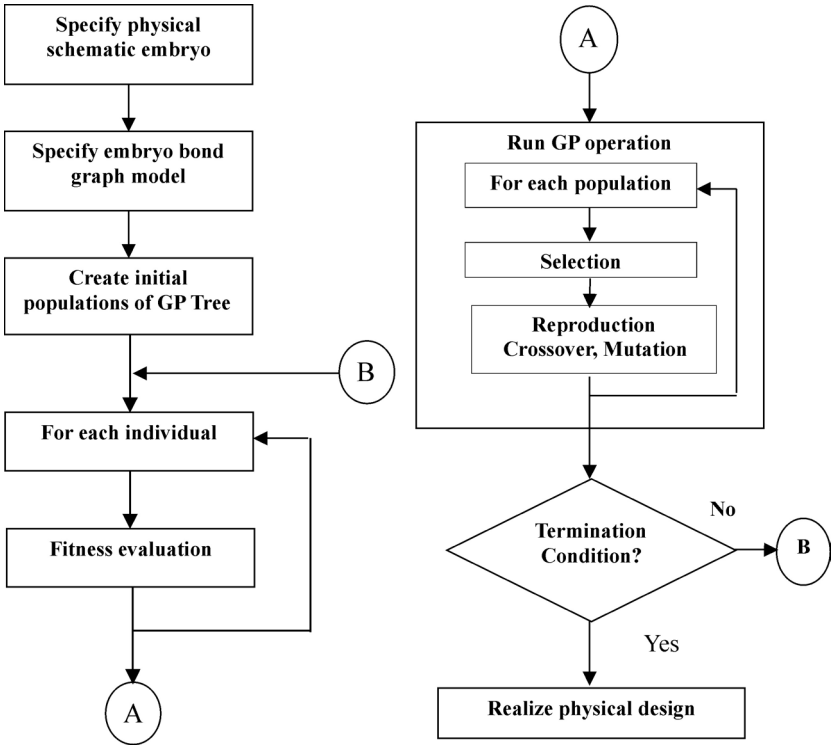

FIGURE 7 Flow chart of the design procedure.

rapidly eliminate further costs for many models that are generated by the genetic programming system, by performing assignment of effort and flow variables and making checks for violations of the appropriate constraints. This simple filtering cuts the evaluation workload dramatically.

2.6 Design Procedure

The flow of the entire algorithm is shown in Figure 7. The user specifies the embryonic physical model for the target system (*i.e.* its interface to the external world, in terms of which the desired performance is specified). After that, an initial population of GP trees is randomly generated. Each GP tree maps to a bond graph tree. Analysis is then performed on each bond graph tree. This analysis consists of two steps – causal analysis and state equation analysis. After the (vector) state equation is obtained, the important dynamic characteristics of the system are sent to the fitness evaluation module and the fitness of each tree is evaluated. For each evaluated and sorted population, genetic operations – selection, crossover, mutation and reproduction – are carried out to seek design candidates with improved quality. The loop of bond graph analysis and GP operation is iterated until a termination condition is satisfied or specified number of iterations is performed. The final step is to instantiate a physical design, replacing the bond graphs with the physical components it represents.

3 CASE STUDIES

Two engineering design problems are investigated as examples to illustrate the utilization and feasibility of this approach. The first example is an analog passive filter design problem, which

shows the efficiency and effectiveness of the approach. The second is the system-level design of a micro-electro-mechanical (MEM) filter. The latter example highlights the steps that must be taken to make the evolved design realizable and manufacturable.

3.1 Analog Passive Filter Design

A filter design problem was used as a test of the approach for evolving electrical circuits with bond graphs. A basic function set was used for the study reported here, in which each junction or component is introduced individually (see Tab. I). The embryo electric circuit and corresponding embryo bond graph model used in the filter design are shown in Figure 8. Converted Matlab routines were used to evaluate the frequency response of the filters created. As Matlab provides many powerful toolboxes for engineering computation and simulation, it facilitates development of source code for the evaluation of GP-evolved designs. In addition, as all individual circuits passed to Matlab code for evaluation are causally valid, the occurrence of singularities is excluded, which enables the program to run continuously without interruption. The fitness function for the analog filter is defined as follows: within the frequency range of interest, uniformly sample 100 points; compare the magnitudes of the frequency response at the sample points with target magnitudes; compute their differences and obtain the sum of squared differences as raw fitness. Then the normalized fitness is calculated according to:

$$\text{Fitness (filter)} = \frac{100}{100 + \sum \text{error}}$$

The GP parameters used for this design problem were as follows:

Number of generations: 100
Population size: 300 in each of thirteen subpopulations and 2500 in each of two subpopulations for HFC
Initial population: half_and_half
Initial depth: 4–6
Max depth: 50

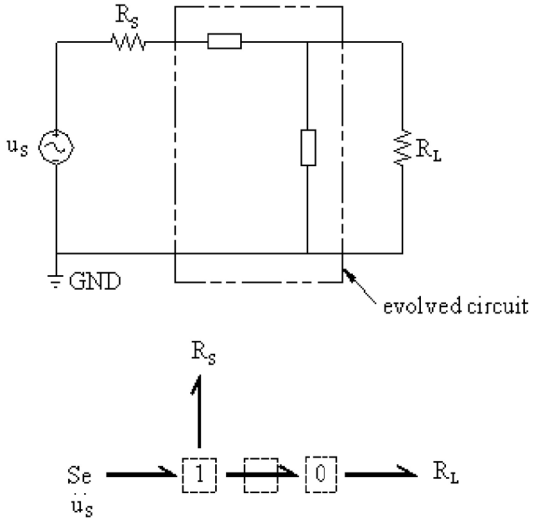

FIGURE 8 Embryo circuit and its bond graph representation.

Max_nodes 5000
Selection: Tournament (size = 7)
Crossover: 0.9
Mutation: 0.3

3.1.1 Results of Analog Passive Filter Design

Results of a high-pass filter design are presented in this article. To illustrate an intermediate step in the evolution of a high-pass filter with a target cutoff frequency of 1000 Hz, the performance of the best design evolved at generation 10 is shown in Figure 9. It is clear that this design is far inferior to that evolved by the end of the run (fewer than 100 generations), as shown in Figure 10. The evolved high-pass filter circuit and bond graph are shown in Figures 11 and 12. Figure 13 shows the fitness history of a typical high-pass filter run.

3.1.2 Discussion

The result of the analog passive high-pass filter design demonstrates both the effectiveness and efficiency of the approach combining bond graphs and GP. It shows that the approach is capable of evolving very satisfactory results in a moderate period of time on a single personal computer. To get this result, the program ran in a P-III 1 GHz for 44.8 min. It took the GP algorithm 100 generations to evolve it. This result is considered to be acquired in an efficient manner because for an evolutionary computation algorithm to evolve designs with similar complexity, it usually takes a much longer time and consumes many more computational resources, typically using clusters of computers [18]. No single factor stands out as the sole reason for this efficiency; it is believed that several factors contribute. The factors are: (1) the bond graph representation

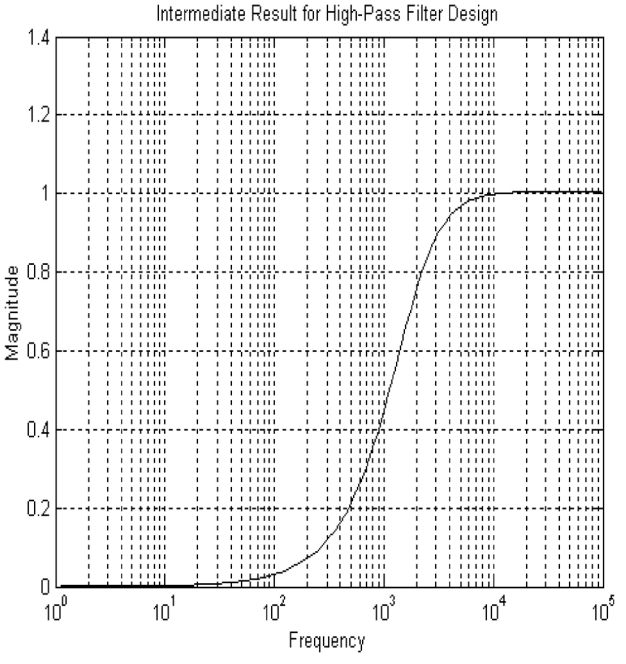

FIGURE 9 Frequency response of an intermediate high-pass filter.

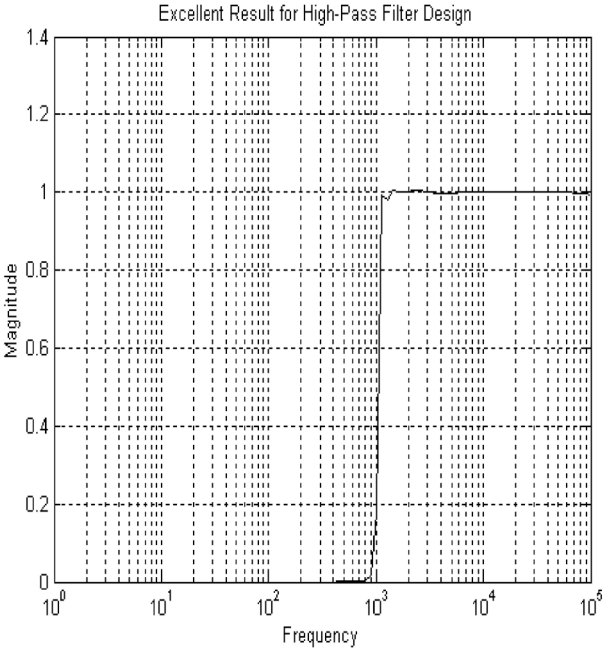

FIGURE 10 Frequency response of the evolved high-pass filter.

of dynamic systems has strong topological expression capability; (2) the genetic operators used promote efficient generation and reconfiguration of bond graph topologies; (3) use of the hierarchical fair competition principle allows search to continue without convergence, assembling elements that contribute to good performance without requiring huge population sizes and numbers of evaluations; (4) causality analysis before evaluating design candidates helps to discard a large volume of improper designs without requiring full evaluations, thus reducing computation time and resources.

3.2 Micro-Electro-Mechanical Filter Design

Automated synthesis of an MEM device, namely, an MEM band pass filter, is taken as an example for this paper. Due to the multi-domain and intrinsically three-dimensional nature of MEMS, their design and analysis is very complicated and requires access to simulation tools with finite element analysis capability, like Conventorware or ANSYS. Computation cost is typically very high, so the first step of modeling and design should use a high-level system model that reduces the number of degrees of freedom from the hundreds and thousands characterizing the meshed 3-D model to as few as possible (resembling in this way the top-down design methods that are so successful in VLSI design) [19]. The model should also have the capability of encompassing multiple energy domains. The bond graph, based on power flow, provides a unified model representation across multiple system domains. In describing the macro behavior of the system, it is also compatible with 3-D numerical simulation, so long as suitable lumping of components can be done to obtain lumped-parameter models. Therefore, the first important step used here in automated synthesis of MEMS is to develop a strategy to automatically generate bond graph models to meet particular design specifications on system-level behaviors. Then in the second or lower level, other numerical optimization approaches [20], as well as evolutionary computation [21], may be used to synthesize custom components from a functionality specification. Figure 14 shows typical structured MEMS

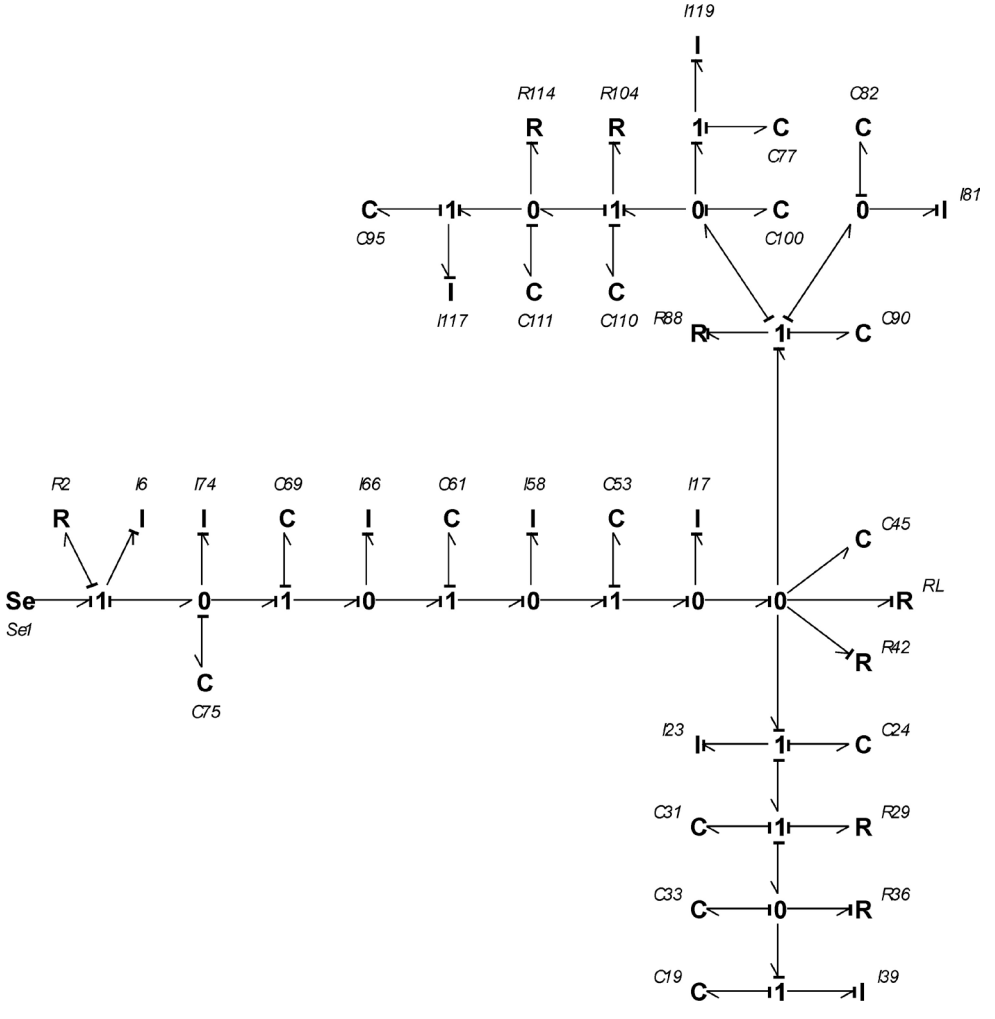

FIGURE 11 Bond graph representation for the evolved high-pass filter.

synthesis procedure, in which the BG/GP approach aims to solve the problem of system-level synthesis in an automatic manner in the first level.

3.2.1 A Lumped-Parameter Model of Micro-Electro-Mechanical Filter Topology

Automated synthesis of micro-mechanical band pass filters is used as an example in this paper [22, 23]. Two popular topologies for these filters, built using surface micromachining, are topologically composed of a series or concatenation of RUs and bridging units (BUs) or RUs and coupling units (CUs). Figures 15 and 16 illustrate the layouts and bond graph representations of two such filter topologies, labeled I and II.

3.2.2 Realizable Function Set

Unlike the designs with basic function sets illustrated with the analog filter above, which impose relatively few topological constraints on the design, MEMS design features relatively few

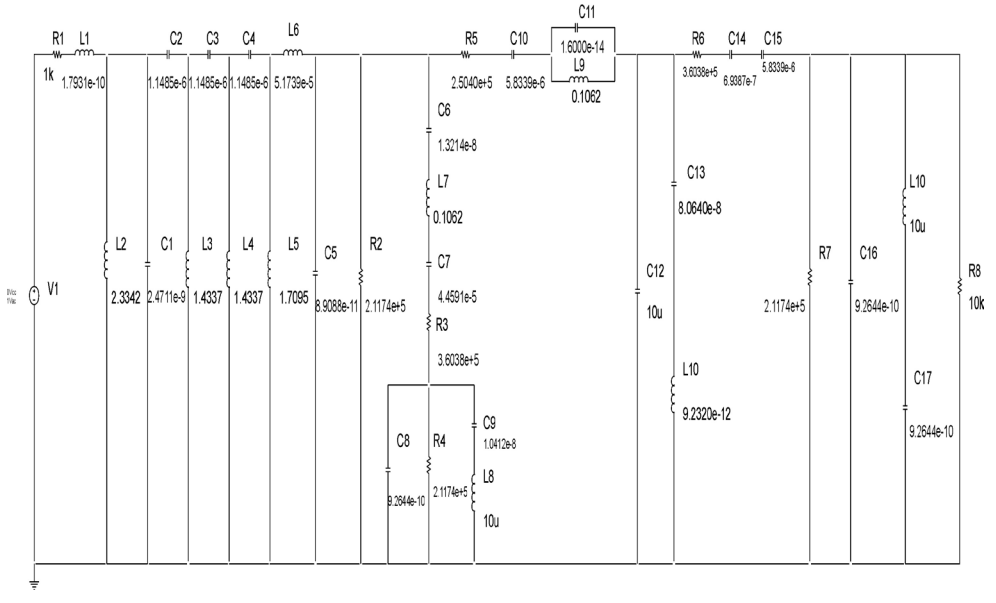

FIGURE 12 Electric circuit for the evolved high-pass filter.

devices in the component library. These devices are typically more complex in structure than those primitive building blocks used in the basic function set. Only evolved designs represented by bond graphs matching the dynamic behavior of those devices which belong to the component library are expected to be manufacturable under current or anticipated technology. Thus, an important and specialized step in MEMS synthesis with the BG/GP approach is to define a

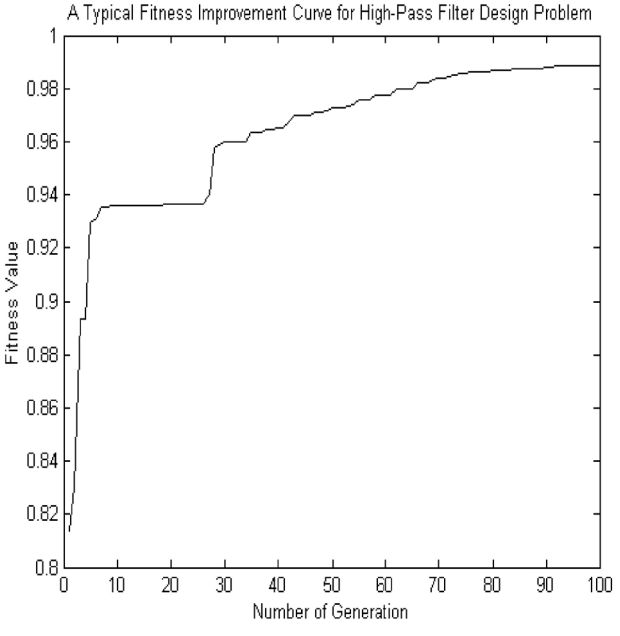

FIGURE 13 Fitness improvement curve of a typical high-pass filter design run.

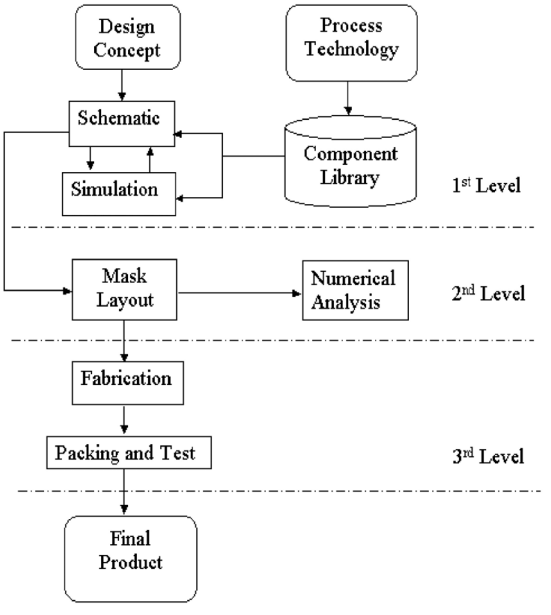

FIGURE 14 Structured MEMS design flow (after Ref. [19]).

realizable function set that, throughout execution, can always produce phenotypes that can be built using existing or anticipated technology.

Analysis of the MEM filter system of Figure 16 from a bond graph viewpoint shows that it is primarily composed of RUs and CUs. The other MEM filter topology shown in Figure 15 includes RUs and BUs. A realizable function set for these design topologies often includes functions from both the basic set and modular set (*i.e.* modules of the level of RUs, BUs, or CUs). In many cases, multiple realizable function sets, rather than only one, can be used to evolve realizable structures for MEMS. This study used the following function sets, along with traditional numeric functions and end operators for creating filter topologies with CUs and RUs.

$$\Re 1 = \{f\_\text{tree}, f\_\text{insert\_}J1, f\_\text{insert\_RU}, f\_\text{insert\_CU}, f\_\text{add\_}C, f\_\text{add\_}R, f\_\text{add\_}I\}$$

$$\Re 2 = \{f\_\text{tree}, f\_\text{insert\_}J1, f\_\text{insert\_RU}, f\_\text{insert\_BU}, f\_\text{add\_}C, f\_\text{add\_}R, f\_\text{add\_}I\}$$

3.2.3 Design Embryo

The MEM filter design problem used the bond graph model shown in Figure 17 as the embryo. The accompanying block diagram indicates that the implementation will accept an electrical signal (voltage) as input and produce a voltage signal as output, but the interior components will be implemented as micromechanical elements.

3.2.4 Adaptive Fitness Function

Filter performance is measured by the magnitude ratio of the frequency response for the voltage across R_L divided by the input voltage u_s. The desired frequency response is unity magnitude

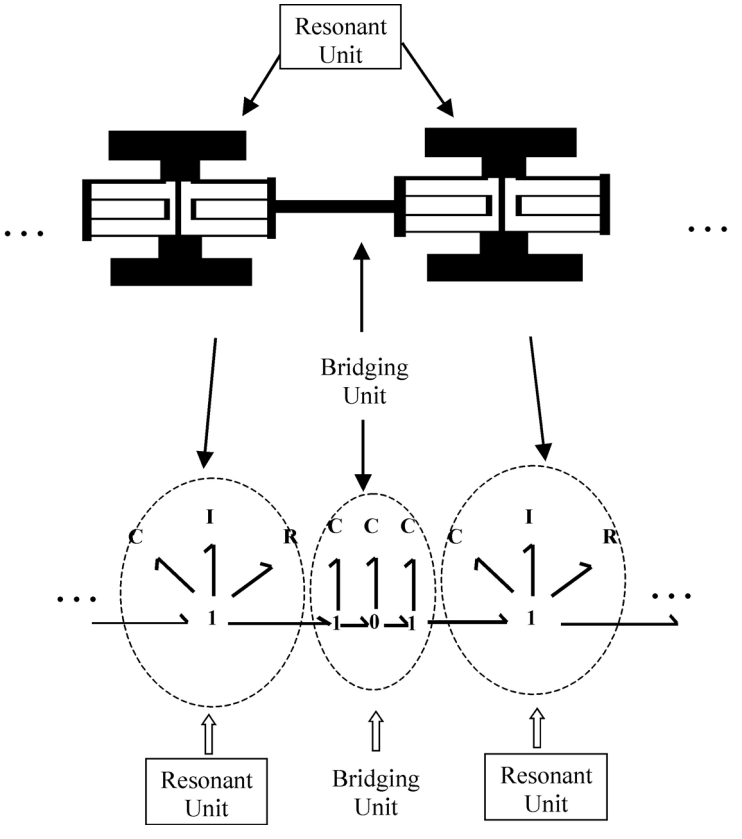

FIGURE 15 Layout of filter topology I: filter is composed of a series of resonator units (RUs) connected by bridging units (BUs).

ratio in the pass band [316–1000 Hz], and zero magnitude ratio outside the pass band. The frequency range of interest is 0.1–100 kHz.

To construct the fitness function evaluator, within the frequency range of interest, 100 points are sampled at equal intervals on a log scale. The magnitudes of the frequency response at the sample points are compared with the target magnitudes. Their differences and a sum of squared differences are computed as raw fitness, defined as $\text{Fitness}_{\text{raw}}$.

If $\text{Fitness}_{\text{raw}} < \text{Threshold}$, change f_{range} to $f^*_{\text{range}} = [f^*_{\text{min}}, f^*_{\text{max}}]$. Usually $f^*_{\text{range}} \subset f_{\text{range}}$. Repeat the above steps and obtain a new $\text{Fitness}_{\text{raw}}$. Then normalized fitness is calculated according to:

$$\text{Fitness}_{\text{norm}} = 0.5 + \frac{\text{Norm}}{(\text{Norm} + \text{Fitness}_{\text{raw}})}.$$

The reason to use adaptive fitness evaluation is that after the population of GP has reached a quite high fitness value as a group, the differences of frequency responses of individuals are to be centered on a more constrained frequency range. In this circumstance, if there are insufficient samplings within this much constrained frequency range, GP may suffer a lack of search pressure as the key factor to push the search forward. Therefore, the frequency range to be heavily sampled is adaptively changed and narrowed. The effect is analogous to narrowing the search window on a small yet most significant area, magnifying it and continuing to search this area with more scrutiny.

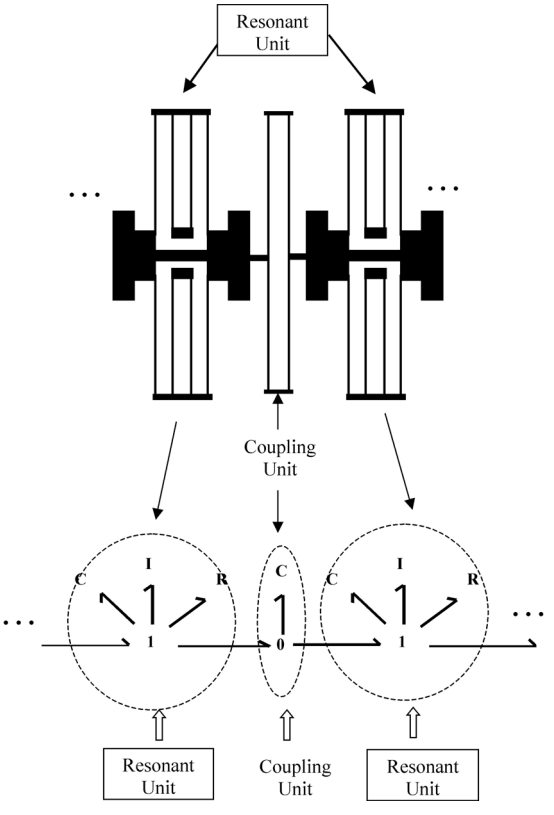

FIGURE 16 Layout of filter topology II: filter is composed of a series of resonator units (RUs) coupled by coupling units (CUs).

3.2.5 Experimental Set-up

The major GP parameters were as shown below

Population size: 500 in each of thirteen subpopulations
Initial population: half_and_half
Initial depth: 4–6
Max depth: 50 Max_nodes 5000
Selection: Tournament (size = 7)
Crossover: 0.9 Mutation: 0.3

3.2.6 Result of Micro-Electro-Mechanical Filter Design

Results of the experiments show the strong topological search capability of genetic programming and the feasibility of the BG/GP approach for finding realizable designs for micro-electro-mechanical filters. Although significant fabrication difficulty is currently presented when fabricating a micro-electro-mechanical filter with more than three resonators, it does not invalidate the research and the topological search capability of the BG/GP approach, considering its potential for exploring more complicated topologies of future MEMS designs and the ever-progressing technology frontiers of MEMS fabrication.

In Figure 18, K is defined as the number of RUs used in the filter topology. It is obvious from the fitness improvement curve that as evolution progresses, the fitness value undergoes continual improvement. It is also interesting that as fitness improves, the value of K also

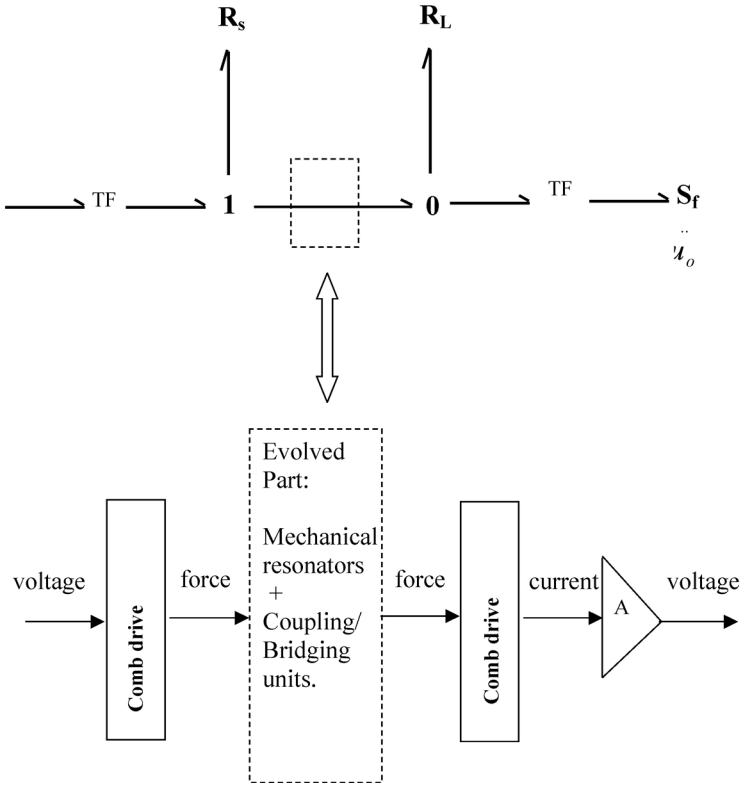

FIGURE 17 Bond graph model of the embryo for MEM filter design and the corresponding block diagram (after Ref. [18]).

becomes larger. This observation is supported by the fact that a higher-order system with more RUs has the potential of better system performance than its low-order counterpart.

The plot of corresponding system frequency responses at generations 27, 52, 117 and 183 is shown in Figure 19. Layouts of a design candidate (evolved part) with three resonators and two BUs as well as its bond graph representation are shown in Figure 20. Notice that the geometry of the resonators may not show the real sizes and shapes of a physical resonator and the layout figure only serves as a topological illustration. The parameters are listed in Table III.

Using the BG/GP approach, it is also possible to explore novel topologies of MEM filter design. In this case, it may not be necessary to use a strictly realizable function set. Instead, a semi-realizable function set may be used to relax the topological constraints with the purpose of finding new topologies not realized before but which are still realizable after careful design. Figure 21 gives an example of a novel topology for an MEM filter design. An attempt to fabricate this kind of topology is being carried out in a university research setting.

3.2.7 Discussion

For design of systems like the MEM filter problem, with strong topological constraints and relatively few topology variations allowed, a major challenge is to define a *realizable* function

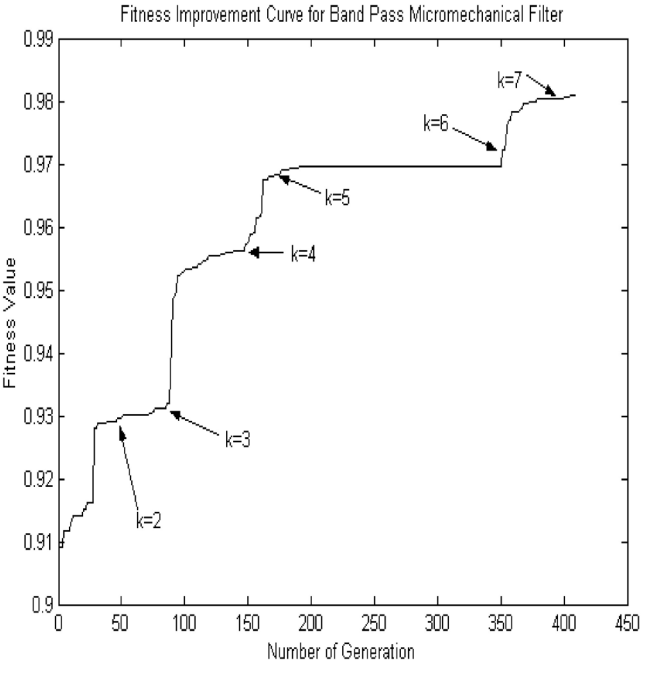

FIGURE 18 Fitness improvement curve of a typical MEM filter design run.

set–one that assures the evolved design can be built using existing or anticipated technologies. Experiments show that a mixture of functions from both a modular function set and a basic function set forms a realizable function set, and that the BG/GP approach, using the hierarchical fair competition principle to increase search efficiency and effectiveness,

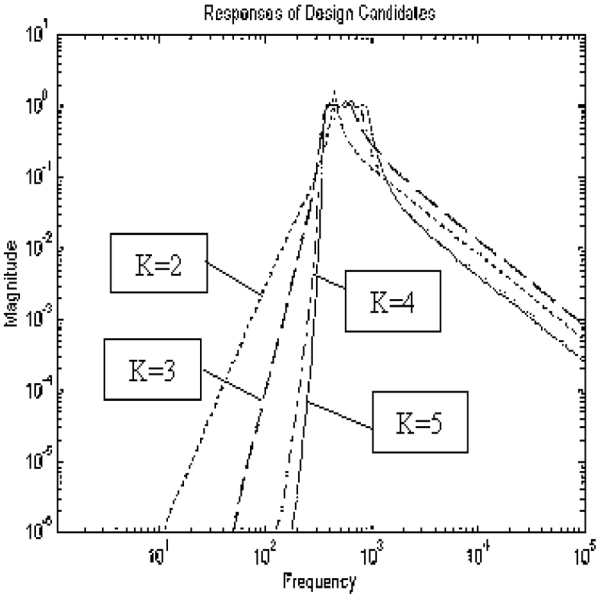

FIGURE 19 Plot of frequency responses of design candidates with different numbers of resonator units. All results are from one GP run.

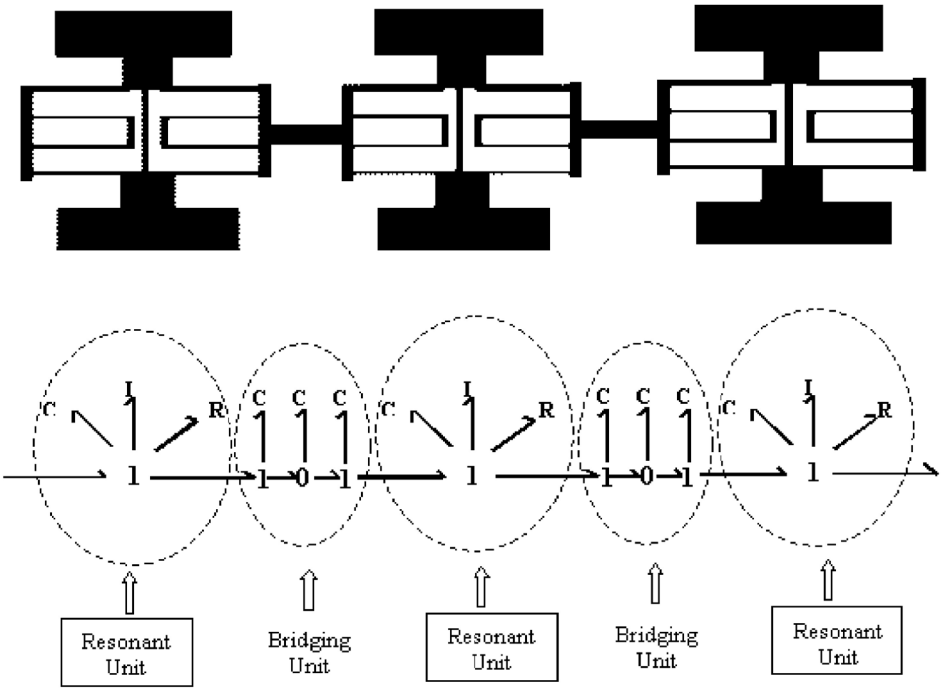

FIGURE 20 Layout and bond graph representation of a design candidate (evolved part) from the experiment with three resonator units coupled with two coupling units.

evolves a variety of designs with different levels of topological complexity that satisfy design specifications.

Many extensions of this research need to be carried out to make the BG/GP approach a more efficient and effective design automation strategy. For example, parameter constraints, in addition to topological constraints, must be taken into account in design automation and optimization of MEMS as well as in many real-world engineering systems. Use of hybrid or memetic algorithms is highly recommended for future exploration, integrating other strong parameter search schemes like evolution strategies, simulated annealing, or other numerical optimization approaches.

TABLE III MEM Filter Element Values.

Parameter	*Value*	*Unit*
C_{x1}	0.0081	F
L_{x1}	0.652	H
R_{x1}	0.139	Ω
C_{ox1}	0.00002737	F
C_{x2}	0.0046	F
L_{x2}	1.589	H
R_{x2}	169.6447	Ω
C_{ox2}	10	F
C_{x3}	0.0024	F
L_{x3}	0.007	H
R_{x3}	0.049	Ω

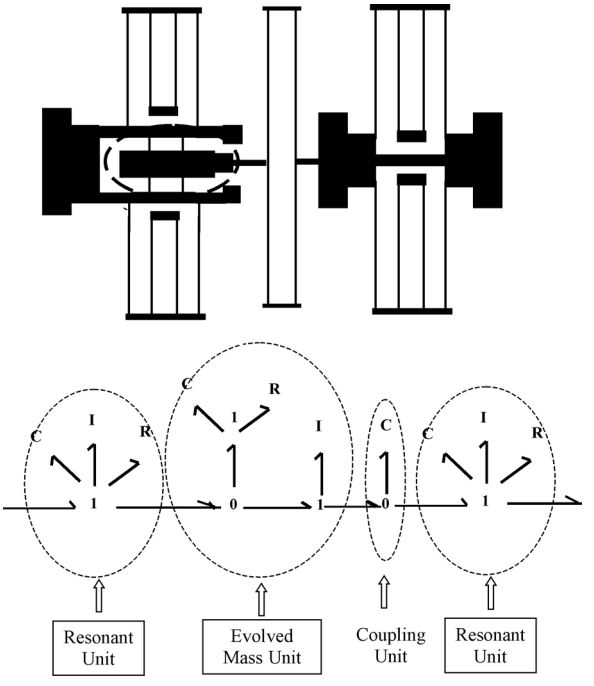

FIGURE 21 A novel topology of MEM filter and its bond graph representation.

4 CONCLUSIONS

This research has explored a new automated approach for synthesizing system-level designs for multi-domain dynamic systems. By taking advantage of genetic programming as a competent search method for designs, and of bond graphs as a representation for dynamic systems, a design environment has been created in which open-ended topological search for system-level models of various classes of engineering systems can be accomplished in an automated manner. The design process is facilitated by the availability of the evolved system-level design candidates, whether the designer wishes to go on to the next step of embodiment of the conceptual designs or, instead, to gain design insight by analyzing the design candidates.

References

[1] Grimbleby, J. B. (2000) Automatic analogue circuit synthesis using genetic algorithms. *IEE Proc.–Circuits Devices Systems*, **147**(6), 319–323.
[2] Lohn, J. D. and Colombano, S. P. (1999) A circuit representation technique for automated circuit design. *IEEE Transactions on Evolutionary Computation*, **3**(3), 205–219.
[3] Koza, J. R. (1994) *Genetic Programming II: Automatic Discovery of Reusable Programs*. The MIT Press, Cambridge, Mass.
[4] Coelingh, H. J., de Vries, T. J. A. and van Amerongen, J. (1998) Automated performance assessment of mechatronic motion systems during the conceptual design stage. *Proc. 3rd Int. Conf. on Adv. Mechatronics*, Okayama, Japan, 472–477.
[5] Hu, J., Goodman, E. D., Seo, K. and Pei, M. (2002) Adaptive Hierarchical Fair Competition (AHFC) model for parallel evolutionary algorithms. *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2002*, New York, pp. 772–779.
[6] Paynter, H. M. (1991) An epistemic prehistory of bond graphs. In *Bond Graphs for Engineers*, (P.C. Breedveld and G. Dauphin-Tanguy (ed.)), Elsevier Science Publishers, Amsterdam, The Netherlands, pp. 3–17.
[7] Karnopp, D. C., Margolis, D. L. and Rosenberg, R. C. (2000) *System Dynamics: Modeling and Simulation of Mechatronic Systems*. Third Edition. John Wiley & Sons, New York.

[8] Rosenberg, R. C. (1993) Reflections on engineering systems and bond graphs. *Trans. ASME J. Dynamic Systems, Measurements and Control*, **115**, 242–251.
[9] Sharpe, J. E. and Bracewell, R. H. (1995) The use of bond graph reasoning for the design of interdisciplinary schemes. *1995 International Conference on Bond Graph Modeling and Simulation*, 116–121.
[10] Tay, E., Flowers, W. and Barrus, J. (1998) Automated generation and analysis of dynamic system designs. *Research in Engineering Design*, **10**(1), 15–29.
[11] Youcef-Toumi, K. (1996) Modeling, design, and control integration: a necessary step in mechatronics. *IEEE/ASME Trans. Mechatronics*, **1**(1), 29–38.
[12] Redfield, R. C. (1999) Bond graphs in dynamic systems designs: concepts for a continuously variable transmission. *1999 International Conference on Bond Graph Modeling and Simulation*, 225–230.
[13] Jamei, M., Mahfouf, M. and Linkens, D. A. (2001) Elicitation and fine-tuning of Mamdani-type fuzzy rules using symbiotic evolution. *European Symposium on Intelligent Technologies, Hybrid Systems and their Implementation on Smart Adaptive Systems (EUNITE 2001)*, Tenerife, Spain.
[14] De Vries, T. J. A. (1994) Conceptual design of controlled electro-mechanical systems–a modeling perspective. *PhD thesis*, University of Twente, Enschede, The Netherlands.
[15] Wang, J. C. and Terpenny, J. (2003) Integrated active and passive mechatronic system design using bond graphs and genetic programming. Late Breaking Paper, *Genetic and Evolutionary Computation Conference GECCO-2003*, Chicago, Illinois.
[16] Holland, J. H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.
[17] Goldberg, D. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Mass.
[18] Koza, J. R., Bennett, F. H., Andre, D., Keane, M. A. and Dunlap, F. (1997) Automated synthesis of analog electrical circuits by means of genetic programming. *IEEE Trans. Evolutionary Computation*, **1**(2), 109–128.
[19] Fedder, G. K. (1996) *Structured Design Methodology for MEMS*. Position Paper for Structured Design Methods for MEMS, Final Report, A Workshop Sponsored by the National Science Foundation, (Erik K. Antonsson, (Ed)), California Institute of Technology.
[20] Zhou, Y. (1998) Layout synthesis of accelerometers. *Thesis for Master of Science*, Department of Electrical and Computer Engineering, Carnegie Mellon University.
[21] Zhou, N., Zhu, B., Agogino, A. and Pister, K. (2001) Evolutionary synthesis of MEMS design. *ANNIE 2001, IEEE Neural Networks Council and Smart Engineering System Design Conference*, St. Louis, MO, **4**(7), 197–202.
[22] Wang, K. and Nguyen, C. T. C. (1999) High-order medium frequency micromechanical electronic filters. *Journal of Microelectromechanical Systems*, **8**(4), 534–556.
[23] De Los Santos, H. J. (1999) *Introduction to Microelectromechanical (MEM) Microwave Systems*. Microelectromechanical System Series, Artech House Publishers, Boston and London.