2005 International Symposium on Nonlinear
Theory and its Applications (NOLTA2005)
Bruges, Belgium, October 18-21, 2005

An Automated Circuit Design Procedure by Means of Genetic Programming

Kenya Jin'no

Dept. of Network and Multi-Media Engineering,
Kanto Gakuin University
1-50-1 Mutsuura-Higashi, Kanazawa, Yokohama, Japan
Phone:+81-45-786-4974 Fax:+81-45-786-4973
Email: kjinno@kanto-gakuin.ac.jp

Abstract—In this article, we discuss an automated circuit design procedure for an analog circuit by means of genetic programming. A novel analog circuit topology can be emerged by our proposed design procedure. In our proposed design procedure, the circuit is represented by a netlist which uses the circuit simulator SPICE. The SPICE netlist can be regarded as a program. Therefore, if we can make an appropriate program, an analog circuit can be designed. In order to create an appropiate program, genetic programming method is proposed. By using the genetic programming, we will propose an automated circuit design procedure for an analog circuit. In this article, we will show two experiments results that our system generates a low-pass filter circuit and high-pass filter circuit.

1. Introduction

Analog electronic circuit design involves a complex problem which is the design of their input-output characteristic and the selection of suitable elements. The design of analog electronic circuits is difficult, and in general, there has been no general automated synthesis procedure. Therefore, some automated synthesis algorithm have been proposed[1]: for example heuristics, knowledge bases, simulated annealing, genetic algorithm, genetic programming, and other algorithm. (please refer Ref.[1].) Koza et al. proposed an automated synthesis procedure that used genetic programming[2],[3]. The synthesis procedure can automatically create parameterized topologies of the desired analog circuits without circuit design theory. The genetic programming creates a kind of program in the LISP programming language in the form of S-expressions. This symbolic expression has a tree structure, but electrical circuits ordinarily have a cyclic graph structure. In order to simulate the created circuit using a circuit simulator,

Figure 1: Objective analog filter

Type	n	Node#1	Node#2	Value

Figure 2: The arrangement of each row of the netlist

the symbolic expression is translated into a netlist which is used on SPICE simulator. The SPICE simulator is the most famous circuit simulator. The procedure, however, uses an infinite number of devices, and therefore the scale of the generated circuits is very large in many cases[2],[3].

Lohn et al. proposed similar automated circuit design procedure that used genetic algorithm[4],[5],[6]. In this procedure, the circuit is represented by a bytecode, and the code must be decoded into the corresponding netlist. This procedure is wanted to minimize computer time during the genetic algorithm runs. However, the decoding process can not be skipped, this algorithm required a large computational capacity.

In a real circuit design process, only the finite number device can be used. In order to emerge a desired analog circuit that uses the minimum number of actual discrete devices, we have proposed an automated synthesis procedure that uses a genetic algorithm for a simple nonlinear analog circuit[7]. Each gene of the genetic algorithm corresponds to a device in the analog circuit. Multiple genes are connected to represent a circuit and these are called a chromosome. The chromosome is represented by a binary code. In order to evaluate generated circuit, the binary code must be translated into SPICE netlist[7]. If the gene is represented by the component of the netlist, the decoding process is not required. In this article, we propose an improved automated analog circuit design procedure. The procedure outputs SPICE netlist directly. The netlist can be regarded as a kind of computer programming language, therefore, our novel procedure can be classified into "Genetic Programming". In order to confirm the performance of our novel procedure, we try to generate some simple analog filters by means of genetic programming.

2. System

In this article, we propose the system can emerge some simple analog filter circuits. The objective simple analog

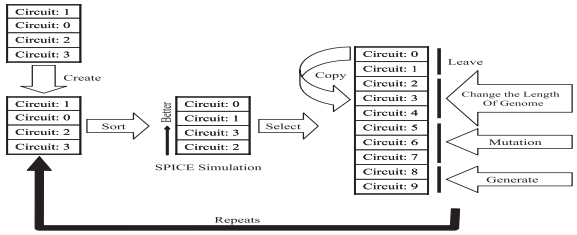

Figure 3: Conceptual diagram of our proposed GA system

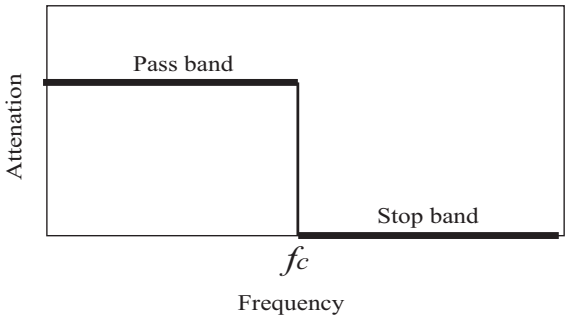

Figure 4: An ideal response of a low-pass filter

filter circuit is represented by a two-port circuit which is composed of two voltage input terminals and two voltage output terminals, as shown in Fig.1.

Since the generated circuit is simulated on SPICE in order to evaluate the circuit properties, the circuit must be described by the SPICE netlist. The SPICE netlist specifies the kind of device, the node number, and the device characteristics. Therefore, the gene describes the SPICE netlist. Each row of the netlist is equivalent to device as shown in Fig.2. The circuits are composed of some devices, then the coressponding devices represents into the netlist. The SPICE netlist is regarded as a gene for our genetic programming algorithm. In Fig.2, ”Type” denotes a kind of device, ”n” means a serial number of the device, ”Node#1” and ”Node#2” denote the connecting node of the device, and ”Value” represents the value of the device.

Our automated synthesis procedure is based on a class of genetic programming. The fundamental searching ability of the genetic programming is equivalent to genetic algorithms. The genetic algorithm can be classified into one of the most effective tools for searching for an optimal solution.

The propose system generates a desired analog circuit topology in the following way.

Step 1 The system generates a population of initial circuits which can be regarded as ”gene” in genetic programming. The initial circuit has a topology and contains some devices which are randomly selected. It is possible to generate effectively if the population has many numbers of circuits. The system, however, requires a large computational capacity to simulate circuit property in the numerical simulation. Therefore, the number of circuits within the population is 10 in the following experiments.

Step 2 We implement an alternation of generations. In the alternation of generation, the system computes the evaluation value of each generated circuit. Each generated circuit is described by the SPICE netlist. We analyze the characteristic of the generated circuit by using a SPICE simulator, and we calculate the evaluation value of each generated circuit by our defined fitness function. By using the evaluation value, it fixes the ranking of the generated circuit.

Step 3 Based on the ranking result, the system generates a new population. The circuits with high evaluation values are left, and the circuits with low evaluation values are eliminated. The new generation circuits are created from the left circuits. At this moment, mutation operations are occurred in some of generated circuits. In this article, the mutation operations are classified into three types. One is the operation to change the length for an elastic gene. This operation is equivalent to fluctuating the number of components in the circuits. When an elastic gene is stretched, the component which is selected randomly, is added. When an elastic gene is shortened, the component which is selected randomly, is deleted. Note that the maximum and the minimum number of components in the circuit is given, so that the above operations occur within the range of possible number of components. In the following our experiments, the maximum number of components is set to 40.

The second mutation operation is changing a kind of the randomly selected component. The value of the changed component sets to the defined initial value. For example, In the case where the changed component is resistor, it makes the value of this resistor 1[$K\Omega$].

The third mutation operation is to change the value of the randomly selected device. It chooses the nearest value within the E24 series.

Repeating the above procedures from ”Step 2” to ”Step 3”, the system can generate a suitable circuit. The conceptual diagram of the above proposed system is illustrated in Fig.3.

Table 1: Initial configuration of experiments

Number of genes in a population	10
Number of elastic genes	3
Number of mutable genes	3
Number of left genes	2
Initial number of components	10
Maximum number of components	40
Minimum number of components	5
Number of alternation of generation	2000

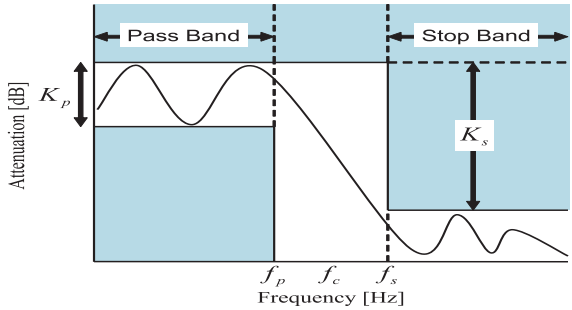

Figure 5: A specification of the desired low-pass filter. f_p = 950[Hz], f_s = 3200[Hz], K_p = 3.0[dB], and K_s = 20.0[dB]

3. Experiments

In this section, we introduce some experimental results. First, a low-pass filter is generated. A low-pass filter has the property that low-frequency excitation signal components down to and including direct current, are transmitted, while high-frequency components, up to and including infinite ones are blocked. The ideal response of a low-pass filter is shown in Fig.4. The range of low frequencies, which are passed, is called the pass band. It extends from 0[Hz] to f_c[Hz]. The highest frequency to be transmitted is f_c[Hz], which is also called the cutoff frequency. Frequencies above cutoff are prevented from passing through the filter and they constitute the filter stopband.

In order to design a low-pass filter, we assume that the desired low-pass filter satisfies the following specification as shown in Fig.5. This design specification is introduced in Ref.[4]. The specification has some margin. The frequency f_p means the highest frequency that the input signal is passed to the output potentially reduced by K_p decibels. The band region which extends from 0[Hz] to f_p[Hz], is called the pass band. The frequency f_s means the lowest frequency that the input signal is decreased by K_s decibels. Above the frequency f_s band region is called the stop band. In this experiment, these parameters set as f_p = 950[Hz], f_s = 3200[Hz], K_p = 3.0[dB], and K_s = 20.0[dB].

In order to evaluate the property of the generated circuit, an evaluation function is defined as

$$Score = -\frac{1}{N}\{\sum_{f_i \in \text{pass band}} (K_p - g(f_i))^2 + \sum_{f_i \in \text{stop band}} (g(f_i) - K_s)^2\}, \quad (1)$$

where f_i represents the frequency of the input signal, $g(f_i)$ represents the gain which is calculated from the SPICE simulation result. K_p and K_s are given specification, and N denotes the number of the observation points.

The initial configuration is given as shown in Table.1. In order to search for a suitable circuit topology efficiently,

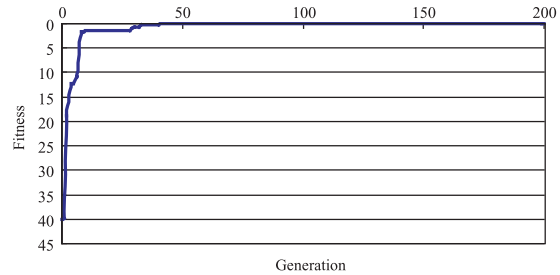

Figure 6: The transition of evaluation value

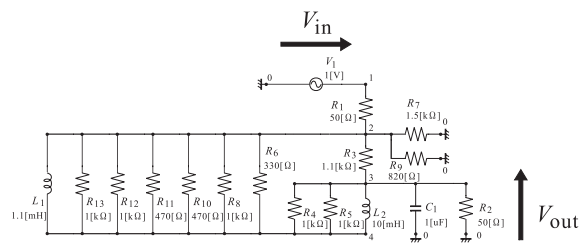

Figure 7: The generated low-pass filter circuit

there should be a small number of left circuits upon the alternation of generations. The operation to change the length of an elastic gene is also for searching for a suitable circuit topology. The number of left genes is 2, and the number of elastic genes is 3. Therefore, a half in the population searches for a suitable circuit topology, and the remainder corresponds to the optimization operation for the circuit. Figure 6 shows the transition of evaluation value. The horizontal axis represents the generation, and the vertical axis denotes the evaluation value.

In this case, the system can discover an excellent circuit within short term. Figure 7 shows the generated analog low-pass filter circuit which indicates the best evaluation value in this experiment. And, its frequency response is shown in Fig.8.

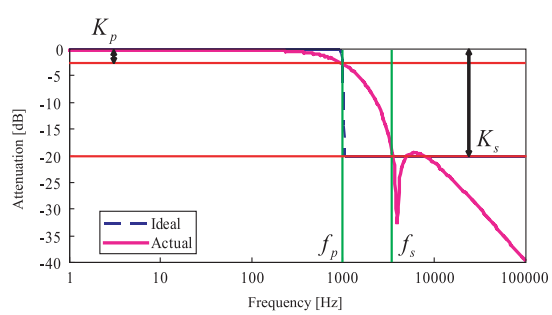

Figure 8: Frequency response of the circuit is shown in Fig.7

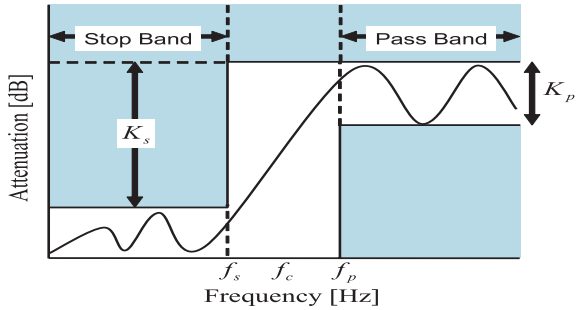

Figure 9: A specification of the desired high-pass filter f_p = 1050[Hz], f_s = 320[Hz], K_p = 3.0[dB], and K_s = 20.0[dB]

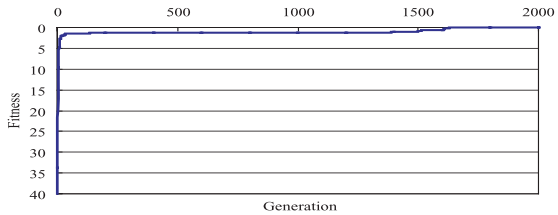

Figure 10: The transition of evaluation value.

Next, we carry out other experiment which a high-pass filter is generated. We assume that the desired high-pass filter satisfies the following specification as shown in Fig.9. In this case, we set the parameters as f_p = 1050[Hz], f_s = 320[Hz], K_p = 3.0[dB], and K_s = 20.0[dB].

Figure 10 shows the transition of evaluation value. Figure 11 shows the generated high-pass filter circuit, and Fig.12 shows its frequency response.

4. Conclusions

We introduced an automated synthesis procedure of an analog circuits. The synthesis procedure can emerge the desired analog circuit topology by using our genetic programming. In this article, we showed two experiments results that our system generated a low-pass filter circuit and high-pass filter circuit. The filter circuit is a kind of fundamental functional circuit. Therefore, an analog circuit expert designer could create a more sophisticated circuit. However, our system can automatically create desired analog circuit without circuit design knowledge.

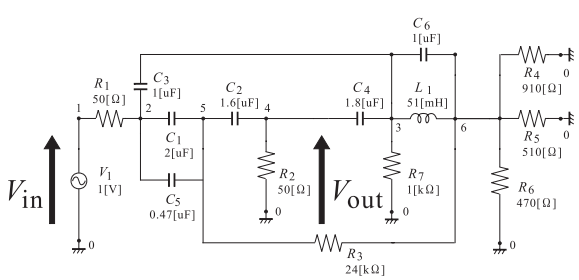

Figure 11: The generated high-pass filter circuit.

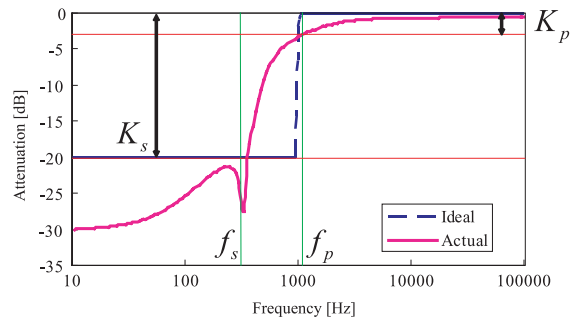

Figure 12: high-pass_circuit050118_frequency.eps

References

[1] J.D. Lohn, "Experiments and Evolving Software Models of Analog Circuits," Communication of the ACM, Special issue on evolvable hardware, , pp. 67-69, April 1999.

[2] J.R. Koza, F. Bennett, D. Andre, M Keane and F.Dunlap, "Automated Synthesis of Analog Electrical Circuits by Means of Genetic Programming," IEEE Trans. Evolutionary Computation, Vol.1, No.2, pp.109-128, July 1997.

[3] J.R. Koza, M.A. Keane, M.J. Streeter, W. Mydlowec, J. Yu and G. Lanza: Genetic Programming IV, Norwell, MA, USA, Kluwer Academic Press, 2003.

[4] J.D. Lohn, and S.P. Colombano, "Automated Analog Circuit Synthesis using a Linear Representation," in Proc. of the Second Int'l Conf. on Evolvable Systems: From Biology to Hardware, Springer-Verlag, Berlin, pp.125-133, 1998.

[5] J.D. Lohn, and S.P. Colombano, "A Circuit Representation Technique for Automated Circuit Design," IEEE Trans. Evolutionary Computation, Vol.3, No.3, pp.205-219, 1999.

[6] J.D. Lohn, S.P. Colombano, G.L.Haith, D.Stassinopoulous, "A Parallel Genetic Algorithm for Automated Electronic Circuit Design," in Proc. of Computational Aeroscience Workshop, NASA Ames Research Centre, Feb. 2000.

[7] H.Ryose, K.Jin'no, and H.Hirose, "Automated Synthesis of Simple Nonlinear Analog Circuits by Means of Genetic Algorithm," Journal of Signal Processing, Vol.8, No.6, pp.529-535, November 2004.