Searching for the Impossible using Genetic Programming

John R. Koza
Section on Medical Informatics
Department of Medicine
Stanford University
Stanford, California 94305
koza@stanford.edu
http://www.smi.stanford.edu/people/koza

Martin A. Keane
Econometrics Inc.
111 E. Wacker Dr.
Chicago, Illinois 60601
makeane@ix.netcom.com

Forrest H Bennett III
Genetic Programming Inc.
Box 1669
Los Altos, California 94023
forrest@evolute.com
http://www.genetic-programming.com

Jessen Yu
Genetic Programming Inc.
Box 1669
Los Altos, California 94023
jyu@cs.stanford.edu

William Mydlowec
Genetic Programming Inc.
Box 1669
Los Altos, California 94023
myd@cs.stanford.edu

Oscar Stiffelman
Computer Science Department
Stanford University
Stanford, California 94305
ozzie@cs.stanford.edu

ABSTRACT

Many potential inventions are never discovered because the thought processes of scientists and engineers are channeled along well-traveled paths. In contrast, the evolutionary process tends to opportunistically solve problems without considering whether the evolved solution comports with human preconceptions about whether the goal is impossible. This paper demonstrates how genetic programming can be used to automate the process of exploring queries, conjectures, and challenges concerning the existence of seemingly impossible entities. The paper suggests a way by which genetic programming can be used to automate the invention process.

We illustrate the concept using a challenge posed by a leading analog electrical engineer concerning whether it is possible to design a circuit composed of only resistors and capacitors that delivers a gain of greater than one. The paper contains a circuit evolved by genetic programming that satisfies the requirement of this challenge as well a related more difficult challenge. The original challenge was motivated by a circuit patented in 1956 for preprocessing inputs to oscilloscopes. The paper also contains an evolved circuit satisfying (and exceeding) the original design requirements of the circuit patented in 1956. This evolved circuit is another example of a result produced by genetic programming that is competitive with a human-produced result that was considered to be creative and inventive at the time it was first discovered.

1 Introduction

In his regular column in *Electronic Design* in 1996, Robert Pease, the legendary analog design engineer and chief scientist at National Semiconductor, posed the question of whether it is possible to design an electrical circuit composed only of resistors and capacitors that delivers a gain greater than one (Pease 1996). Pease observed that most electrical engineers would say that it is "absurd" to try to build such a circuit using only resistors and capacitors.

> "Resistors are passive. How can you take a network of Rs and Cs and generate a gain of greater than one? That's impossible!"

Pease then focused on a circuit that was patented in 1956 by George Philbrick, one of the early pioneers of analog circuit design. The Philbrick circuit, consisting of only three resistors and three capacitors, was motivated by the need at the time for a circuit to preprocess the input to a cathode-ray oscilloscope (Philbrick 1956). However, as Pease observed in 1996, this RC circuit has a surprising characteristic concerning gain that is unrelated to the circuit's intended purpose. As Pease showed in 1996, if the output of the Philbrick circuit is attached to a unity-gain follower (such as op amp with a gain of +1.00) and the output of the follower is, in turn, fed back to the input of the Philbrick circuit, the resulting circuit is a phase-shift oscillator. According to the Nyquist criterion, a circuit can only oscillate if the gain around the loop is greater than 1. However, since the unity-gain op amp has a gain of only 1, then, by inference, Philbrick's RC circuit must have a gain of greater than 1. In fact, it does.

Figure 1 shows the RC circuit that Philbrick patented in 1956. In the figure, in incoming signal VSRC passes into the network of resistors and capacitors. The voltage is probed at probe point V.

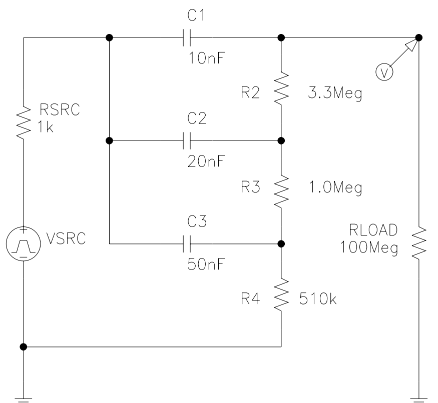

Figure 1 Philbrick 1956 circuit consisting of three capacitors and three resistors.

Figure 2 shows the output voltage measured at point V of the circuit of figure 1 for a 1-volt input signal. The horizontal axis represents six decades of frequencies between 1 milli-Hertz and 1,000 Hertz (Hz). As can be seen, this RC circuit delivers a gain of greater than 1.0 for frequencies between approximately 2.0 Hz and 20.0 Hz. The voltage peaks at 1.19 volts at 4.64 Hz.

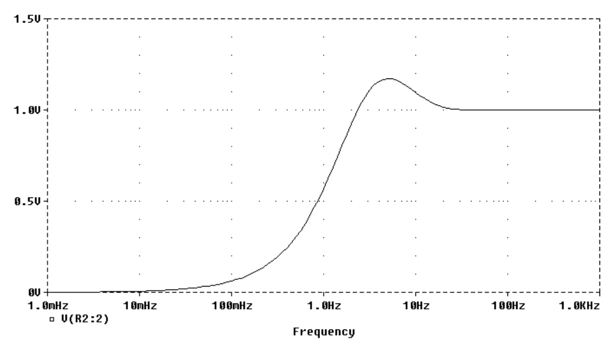

Figure 2 Output of 1956 Philbrick circuit.

The question arises as to whether it is possible to automate the process of exploring challenges, such as the one above posed by Pease above, concerning the existence of seemingly impossible constructions.

We posed Pease's challenge informally to a number of electrical engineers (all of whom, like all the authors of this paper, initially said such a circuit was impossible). Interestingly, once they were made aware of the Philbrick 1956 circuit, all were able to construct a reasonable explanation as to why the Philbrick circuit works. The observed counter-intuitive voltage multiplication of the Philbrick circuit comes about as a result of a phase shift. In particular, the output of the Philbrick circuit is the difference of a 1-volt (peak amplitude) input signal of the form $\sin 2\pi ft$ and another signal of the form $\sin(2\pi ft + \Delta)$, where Δ represents a phase shift. Although the particular circuit in Philbrick's patent delivers a gain of 1.19, it is clear that it is possible to construct an RC circuit in a similar way that delivers a gain approaching 2. However, a new query arises as a result of this line of reasoning. Is it possible to construct an RC circuit can deliver a gain of greater than 2? Again, several electrical engineers and the authors of this paper said such a circuit was impossible. For this new query, we did not have foreknowledge of the outcome from the Pease article.

This paper demonstrates how genetic programming can be can be used to automate the process of exploring queries concerning the existence of seemingly impossible electrical circuits. In particular, the paper demonstrates the counter-intuitive result that an electrical circuit composed only of resistors and capacitors can deliver a gain of 2.24.

Section 2 provides background on the application of genetic programming to the automatic creation of electrical circuits. Section 3 describes the preparatory steps necessary to apply genetic programming to the query concerning the existence of an RC circuit with a gain of greater than one. Section 4 shows an evolved RC circuit with a gain of 2.24. Section 5 describes the preparatory steps for evolving a circuit with the same specifications as the circuit that Philbrick patented in 1956. Section 6 shows an evolved circuit that satisfies the specifications of Philbrick's circuit.

2 Background on Genetic Programming

Genetic programming is an extension of the genetic algorithm (Holland 1975). Genetic programming automatically creates computer programs to solve problems. Genetic programming is described in Koza 1992; Koza and Rice 1992; Koza 1994a, 1994b; Banzhaf, Nordin, Keller, and Francone 1998; Langdon 1998; Kinnear 1994; Angeline and Kinnear 1996; Spector, Langdon, O'Reilly, and Angeline 1999; Koza, Goldberg, Fogel, and Riolo 1996; Koza, Deb, Dorigo, Fogel, Garzon, Iba, and Riolo 1997; Koza, Banzhaf, Chellapilla, Deb, Dorigo, Fogel, Garzon, Goldberg, Iba, Riolo 1998; and Banzhaf, Poli, Schoenauer, and Fogarty 1998; and Poli, Nordin, Langdon, and Fogarty 1999.

It has been recently demonstrated genetic programming is capable of synthesizing the design of a wide variety of analog electrical circuits (Koza, Bennett, Andre, Keane, and Dunlap 1997; Koza, Bennett, Andre, and Keane 1999a, 1999b). Since genetic programming is a probabilistic process that is not encumbered by the preconceptions that often channel human thinking down familiar paths, it often creates novel designs. In fact, nine of the analog circuits that were evolved in *Genetic Programming: Darwinian Invention and Problem Solving* (Koza, Bennett, Andre, and Keane 1999a) were previously patented. Specifically, genetic programming rediscovered the Darlington emitter-follower transistor circuit (patented by Sidney Darlington of American Telephone and Telegraph in 1952), the circuit that is now known as the "constant *K*" ladder filter (patented by George Campbell in 1917), the "*M*-derived half section" for a filter (patented by Otto Zobel in 1925), the elliptic filter topology (patented by Wilhelm Cauer between 1934 and 1936), and the crossover filter (patented by Otto Zobel in 1925). In addition, genetic programming has successfully evolved several different high-gain amplifiers, several different computational circuits, an electronic thermometer, and a voltage reference circuit (all of which were covered by one or more patents in the past 30 years).

3 Preparatory Steps for Evolving RC Circuit with Gain Greater than One

Seven major preparatory steps are required to apply genetic programming to a problem of circuit synthesis: (1) identify the initial circuit (test fixture and embryo) of the developmental process, (2) determine the architecture of the circuit-constructing program trees, (3) identify the primitive functions of the program trees, (4) identify the terminals of the program trees, (5) create the fitness measure, (6) choose control parameters, and (7) determine the termination criterion and method of result designation.

3.1 Initial Circuit

An electrical circuit can be created by genetic programming by means of a developmental process. This developmental process entails the execution of a circuit-constructing program tree that contains various component-creating, topology-modifying, and development-controlling functions. An initial circuit consisting of an embryo and a test fixture is the starting point of the developmental process for transforming a program tree in the population into a fully developed electrical circuit. The embryo contains at least one modifiable wire. The test fixture is a fixed (hard-wired) substructure composed of nonmodifiable wires and nonmodifiable electrical components. The test fixture provides access to the circuit's external input(s) and permits probing of the circuit's output. A test fixture has one or more ports that enable an embryo to be embedded into it. An embryo has one or more ports that enable it to communicate with the test fixture in which it is embedded. All development originates from the modifiable wires.

Figure 3 shows a one-input, one-output initial circuit consisting of an embryo embedded in a test fixture. The embryo consists of two modifiable wires `Z0`, and `Z1`. The test fixture has an incoming signal source `VSOURCE`, a source resistor `RSOURCE`, a nonmodifiable wire `ZOUT`, a voltage probe point `VOUT` (the output of the overall circuit), a variable output load called `LOAD`, and a nonmodifiable wire `ZGND` providing a connection to ground.

3.2 Program Architecture

Since there is a result-producing branch in the program tree for each modifiable wire in the embryo, the architecture of each circuit-constructing program tree has two result-producing branches.

3.3 Function Set

The function set, $\mathcal{F}_{\text{CCS}}$, for each construction-continuing subtree is

$\mathcal{F}_{\text{CCS}}$ = {`R`, `C`, `SERIES`, `PARALLEL0`, `PARALLEL1`, `FLIP`, `NOP`, `PAIR_CONNECT_0`, `PAIR_CONNECT_1`}.

All functions in this section are described in detail in Koza, Bennett, Andre, and Keane 1999. Briefly, the `R` and `C` functions are component-creating functions that insert a resistor or capacitor (respectively) into a developing circuit and that establish the numerical value of the inserted component. The `SERIES` and the two `PARALLEL` functions modify the topology of the developing circuit by performing a series or parallel (respectively) division. The `FLIP` function reverses the polarity of a component. The `NOP` (No operation) function is a development-controlling function The two `PAIR_CONNECT` functions provide a way to connect two (usually distant) points in the developing circuit.

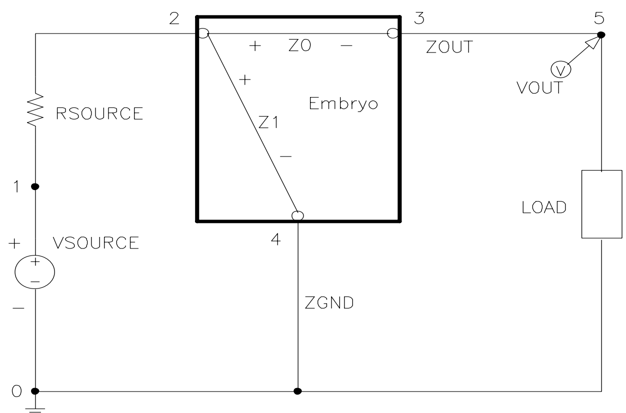

Figure 3 One-input, one-output initial circuit with two modifiable wires, source resistor, and variable load.

3.4 Terminal Set

The initial terminal set, $\mathcal{T}_{\text{CCS}}$, for each construction-continuing subtree is

$\mathcal{T}_{\text{CCS}}$ = {`END`, `SAFE_CUT`}.

Briefly, the development-controlling `END` function makes the modifiable wire or modifiable component with which it is associated non-modifiable (thereby ending a particular developmental path). The `SAFE_CUT` function causes the highlighted component to be removed from the circuit in a way that preserves the validity of the circuit.

The initial terminal set, $\mathcal{T}_{\text{aps}}$, for each arithmetic-performing subtree consists of

$\mathcal{T}_{\text{aps}}$ = {$\Re$}.

$\Re$ represents floating-point constants from –1.0 to +1.0.

The function set, $\mathcal{F}_{\text{aps}}$, for each arithmetic-performing subtree is,

$\mathcal{F}_{\text{aps}}$ = {`+`, `-`}.

3.5 Fitness Measure

The evaluation of each individual circuit-constructing program tree in the population begins with its execution. The execution progressively applies the functions in the program tree to the embryo of the circuit, thereby creating a fully developed circuit. A netlist is created that identifies each component of the developed circuit, the nodes to which each component is connected, and the value of each component. The netlist becomes the input to our modified version of the 217,000-line SPICE (Simulation Program with Integrated Circuit Emphasis) simulation program (Quarles, Newton, Pederson, and Sangiovanni-Vincentelli 1994). SPICE then determines the behavior of the circuit.

The output voltage VOUT is measured in the frequency domain. SPICE is instructed to perform two AC analyses on each circuit using different output loads. In the first simulation, LOAD has infinite resistance (i.e., an open circuit). In the second simulation, LOAD consists of a 10 mega-ohm resistor in parallel with a 6 pico-farad capacitor (i.e., a load resembling that of an oscilloscope). In each case, only the voltage for 1,000 Hz is used in computing fitness. The fitness of a circuit is

$$1/(1+ v_{\text{out-infinite}}) + 1/(1+ v_{\text{out-oscilloscope}})$$

where $v_{\text{out-infinite}}$ is the voltage at 1,000 Hz for the open circuit and $v_{\text{out-oscilloscope}}$ is the voltage at 1,000 Hz for the load resembling that of an oscilloscope.

Circuits that cannot be simulated by SPICE receive a high penalty value of fitness (10^8).

3.6 Control Parameters

The population size, *M*, is 660,000. A maximum size of 800 points (functions and terminals) was established for each branch of each circuit-constructing program tree. Other control parameters were the ones that are used previously for the lowpass filter problem in chapter 25 and appendix D of Koza, Bennett, Andre, and Keane 1999.

3.7 Termination

Since the maximum achievable gain for an RC circuit was not known in advance, the run was manually terminated when no further progress appeared likely in the run.

3.8 Parallel Implementation

This problem was run on a home-built Beowulf-style (Sterling, Salmon, Becker, and Savarese 1999) parallel cluster computer system consisting of 66 processing nodes (each containing a 533-MHz DEC Alpha microprocessor and 64 megabytes of RAM) arranged in a 6 × 11 toroidal mesh. The system has a DEC Alpha computer as host. The processing nodes are connected with 100 megabit-per-second Ethernet. The processing nodes and host use the Linux operating system. The distributed genetic algorithm was used with a population size of Q = 10,000 at each of the D = 66 demes (semi-isolated subpopulations). Generations are asynchronous on the nodes. On each generation, four boatloads of emigrants, each consisting of B = 2% (the migration rate) of the node's subpopulation (selected probabilistically on the basis of fitness) were dispatched to each of the four toroidally adjacent processing nodes. Details are found in Andre and Koza 1996; Koza, Bennett, Andre, and Keane 1999; and Bennett, Koza, Shipman, and Stiffelman 1999).

4 Results for Evolving RC Circuit with Gain Greater than One

The best circuit-constructing program of the 660,000 programs of generation 0 has a fitness of 0.956.

In generation 15, a circuit (figure 4) consisting of three resistors and three capacitors is evolved with a fitness of 0.913 and that produces a gain of 1.19. This circuit is topologically different from the circuit in the 1956 patent while delivering approximately the same gain.

In generation 927, a circuit (figure 6) consisting of 38 resistors and 35 capacitors is evolved with a fitness of 0.622 and that produces a gain of 2.24. Figure 7 shows the behavior in the frequency domain of this circuit.

5 Preparatory Steps for Evolving Circuit of 1956 Philbrick Patent

5.1 Initial Circuit

Figure 8 shows a one-input, one-output initial circuit consisting of an embryo embedded in a test fixture. The embryo consists of one modifiable wire Z0. The test fixture consists of an incoming signal source VSOURCE, a 1,000 Ω source resistor, a nonmodifiable wire ZOUT, a voltage probe point VOUT, a 100 mega-ohm load resistor, and a nonmodifiable wire ZGND connecting to ground.

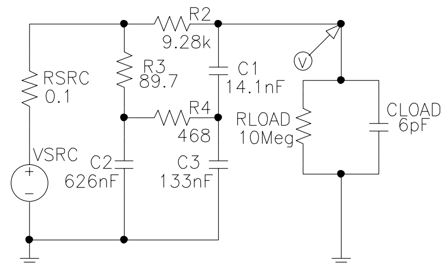

Figure 4 Best circuit from generation 15.

Figure 5 shows the behavior in the frequency domain of the best circuit from generation 15.

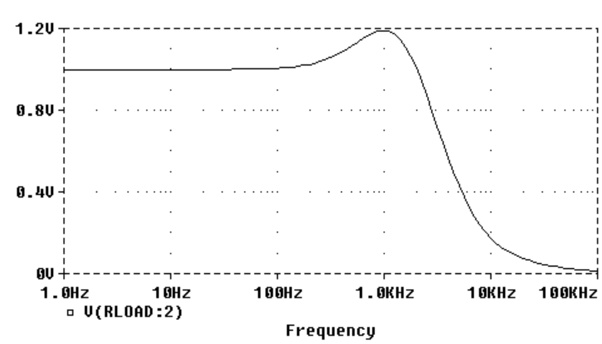

Figure 5 Behavior in frequency domain of best circuit from generation 15.

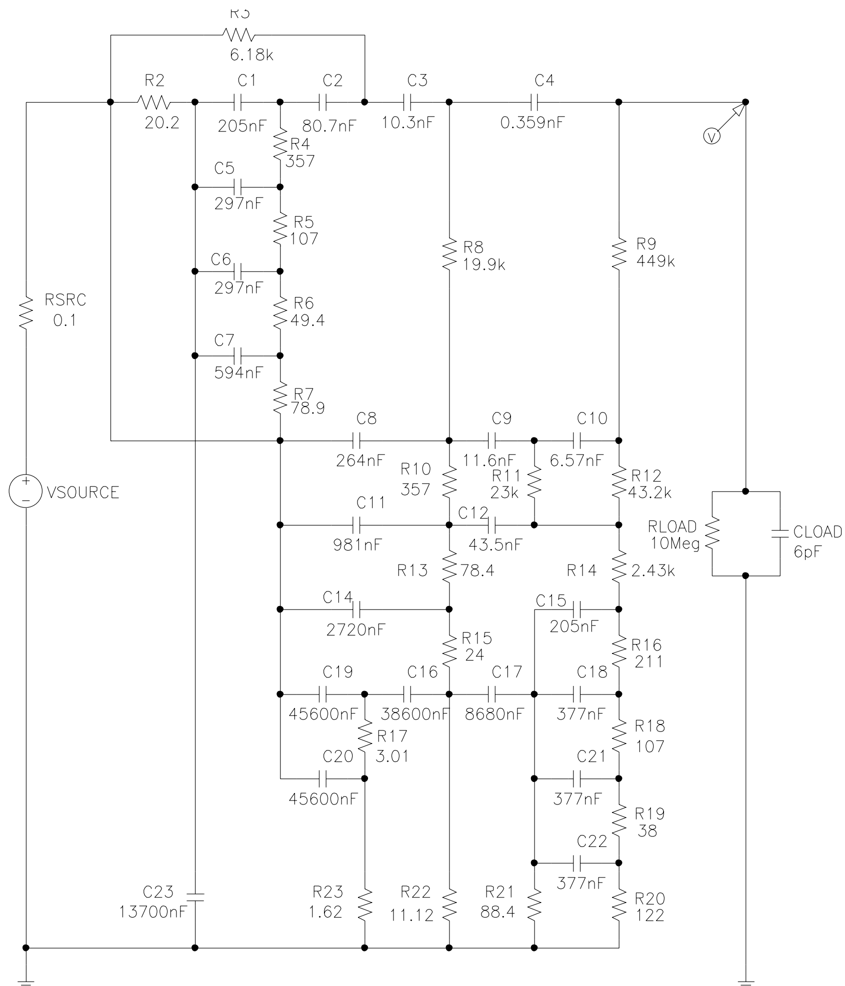

Figure 6 Best circuit from generation 927.

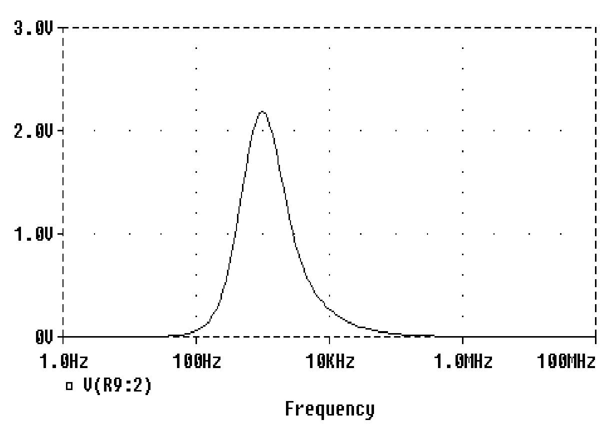

Figure 7 Behavior in frequency domain of best circuit from generation 927.

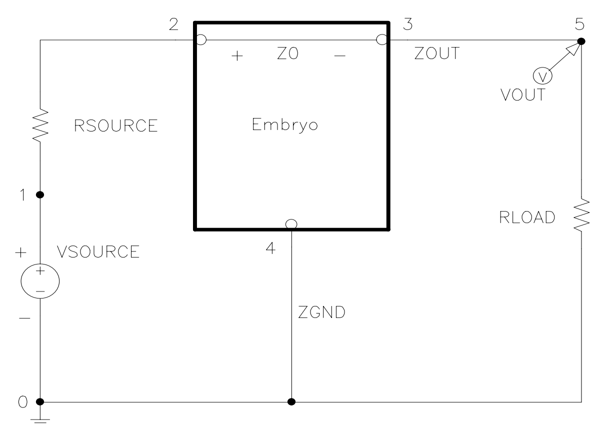

Figure 8 One-input, one-output initial circuit with two modifiable wires.

5.2 Program Architecture

Since there is a result-producing branch in the program tree for each modifiable wire in the embryo, the architecture of each circuit-constructing program tree has one result-producing branch.

5.3 Function Set

The function set, $\mathcal{F}_{\text{CCS}}$, for each construction-continuing subtree is

$\mathcal{F}_{\text{CCS}}$ = {R, C, SERIES, PARALLEL0, PARALLEL1, FLIP, NOP, PAIR_CONNECT_0, PAIR_CONNECT_1, RETAINING_THREE_GROUND0, RETAINING_THREE_GROUND1}.

All functions in this section are described in detail in Koza, Bennett, Andre, and Keane 1999. Briefly, the R and C functions are component-creating functions that insert a resistor or capacitor (respectively) into a developing circuit and that establish the numerical value of the inserted component. The SERIES and the two PARALLEL functions modify the topology of the developing circuit by performing a series or parallel (respectively) division. The FLIP function reverses the polarity of a component. The NOP (No operation) function is a development-controlling function The two PAIR_CONNECT functions provide a way to connect two (usually distant) points in the developing circuit. The RETAINING_THREE_GROUND functions provide a way to connect a point in the developing circuit to ground.

5.4 Terminal Sets

The initial terminal set, $\mathcal{T}_{\text{CCS}}$, for each construction-continuing subtree is

$\mathcal{T}_{\text{CCS}}$ = {END, SAFE_CUT}.

Briefly, the development-controlling END function makes the modifiable wire or modifiable component with which it is associated non-modifiable (thereby ending a particular developmental path). The SAFE_CUT function causes the highlighted component to be removed from the circuit in a way that preserves the validity of the circuit.

The initial terminal set, $\mathcal{T}_{\text{aps}}$, for each arithmetic-performing subtree consists of

$\mathcal{T}_{\text{aps}} = \{\Re\}$.

$\Re$ represents floating-point constants from –1.0 to +1.0.

The function set, $\mathcal{F}_{\text{aps}}$, for each arithmetic-performing subtree is,

$\mathcal{F}_{\text{aps}}$ = {+, -}.

5.5 Fitness Measure

The circuit patented by Philbrick was intended to preprocess an analog signal that was to be fed into an oscilloscope. As Philbrick (1956) stated,

> "This invention relates to an electric filter network, and more particularly to a delayed-recovery, high-pass filter network, which transmits the early portion of a transient voltage signal substantially without distortion but the output of which thereafter relatively rapidly recovers to a quiescent value of zero voltage when the impressed signal becomes quiescent at any voltage.
>
> "Filter networks of this type frequently are required in electronic equipment for A. C. coupling, pulse forming and shapping, differentiating, etc. For example, in cathode-ray oscilloscopes it often is desirable to display high-frequency transients without distortion and thereafter to return the horizontal trace relatively rapidly to zero value when the input signal returns to a steady state value. This necessitates a special input circuit.
>
> Prior to the present invention no simple filter was available to accomplish this result. In the past it has been customary to use a simple series-capacitor, shunt-resistor, high-pass filter network for this purpose. When this conventional type of filter network is used in such circuits, it is subject to certain disadvantages. If high-frequency transients are to be transmitted with negligible distortion its recovery time usually is entirely too long for practical purposes or, if made to have a recovery time of practical length, considerable distortion is introduced into the transient being transmitted.
>
> "Accordingly, it is an object of the present invention to provide a filter network having a transmission characteristic such that initial high-frequency transients of the input signal are transmitted substantially without distortion followed after a predetermined time-delay by a relatively rapid return of the output voltage to zero irrespective of the actual magnitude of the quiescent value of the input signal."

Our fitness measure is based on the closeness of the behavior of a candidate circuit, in both the frequency and time domain, and the behavior of the circuit in Philbrick's patent. The first portion of the fitness measure is a weighted sum of the discrepancies between the candidate circuit's behavior in the frequency domain (an AC analysis in SPICE) and the actual behavior of the circuit in Philbrick's patent. The candidate circuit is simulated at 121 frequency values in an interval of six decades of frequency values between 1 millihertz and 1,000 Hz. This portion of the fitness measure is the sum, over the 121 fitness cases, of the absolute weighted deviation between the actual value of the voltage that is produced by the circuit at the probe point VOUT and the target value for voltage. The absolute difference between 1 volt (the desired voltage between 1 Hz and 1,000 Hz) and the actual output voltage for the 61 points between 1 Hz and 1,000 Hz is weighted by 1.0 if the difference is above 970 millivolts and otherwise weighted by 10. The absolute difference between 0 volts (the desired voltage at 1 millihertz) and the actual output voltage at 1 millihertz is weighted by 60.0. Discrepancies for the remaining 59 points are ignored.

The second portion of the fitness measure is a weighted sum of the discrepancies between the candidate circuit's behavior in the time domain (a transient analysis in

SPICE) and the behavior of the circuit in Philbrick's patent. The circuit is simulated over 121 time steps in an interval between 0 and 120 milliseconds. This portion of the fitness measure is the sum, over the 121 fitness cases, of the absolute weighted deviation between the actual value of the voltage that is produced by the circuit at the probe point VOUT and the target value for voltage. The absolute difference between 1 volt (the desired voltage between 0 and 10 milliseconds) and the actual output voltage for the 11 points between 0 and 10 milliseconds is weighted by 60 if the difference is within 30 millivolts of 1 volt and otherwise weighted by 600. The absolute differences between 0 volts (the desired voltage between 60 milliseconds and 120 milliseconds) and the actual output voltage for the 61 points are weighted by 1 if the difference is within 300 millivolts of 0 volts and otherwise weighted by 10. The discrepancies for the remaining 49 points are ignored.

The occasional circuit that cannot be simulated by SPICE receives a high penalty value of fitness (10^8).

5.6 Control Parameters

The population size, *M*, is 660,000. A maximum size of 800 points (functions and terminals) was established for the one result-producing branch of each circuit-constructing program tree. Other control parameters were the ones that are used previously for the lowpass filter problem in chapter 25 and appendix D of Koza, Bennett, Andre, and Keane 1999.

6 Results for Evolving Circuit of 1956 Philbrick Patent

The best circuit from generation 0 consists of four resistors and three capacitors (not counting the components in the test fixture). It has a fitness of 960.13. The portion of the fitness measure pertaining to the frequency domain is 12.67 and the portion pertaining to the time domain is 947.13.

The best circuit from generation 39 (figure 9) consists of six resistors and six capacitors (not counting the components in the test fixture). This circuit has a fitness of 665.55. The portion of the fitness measure pertaining to the frequency domain is 49.93 and the portion pertaining to the time domain is 615.62.

The above run of genetic programming was continued and a somewhat better circuit appeared in generation 214. This evolved circuit was considerably larger than the best circuit of generation 39. This evolved circuit from generation 214 consists of 12 resistors and 15 capacitors. The fitness of this circuit is 663.03 (47.64 for the frequency domain and 615.39 for the time domain). Since the fitness of the larger circuit is only 0.4% better than that of the best circuit from generation 39, we now focus our attention on the best circuit from generation 39.

Figure 10 shows the behavior in the frequency domain of the best circuit from generation 39.

Figure 11 compares the time-domain behavior of the Philbrick circuit (without boxes) and the best circuit from generation 39 (with boxes).

The requirement that Philbrick (1956) established for, and satisfied with, his patented circuit was to transmit "the early portion of a transient voltage signal substantially without distortion" for approximately 10 milliseconds and to recover "to a quiescent value of zero voltage" after 100 milliseconds. As can be seen in figure 11, the first 10 milliseconds of the transient voltage signal of the evolved best circuit from generation 39 (bottom curve with boxes) is virtually the same as that for Philbrick's circuit (top curve without boxes). After 100 milliseconds, the transient voltage signal of the evolved best circuit from generation 39 (with boxes) is suppressed to a greater degree than that of Philbrick's circuit (without boxes). In other words, the genetically evolved circuit from generation 39 satisfies Philbrick's own requirement to a greater degree than Philbrick's patented circuit.

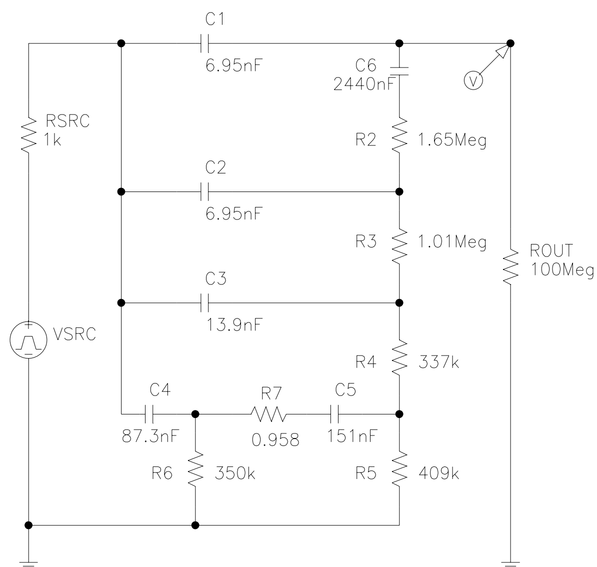

Figure 9 Best Evolved circuit from generation 39 that satisfies specifications of Philbrick circuit.

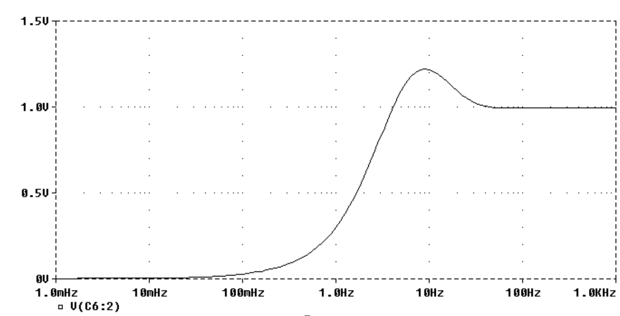

Figure 10 Frequency domain behavior of best circuit from generation 39.

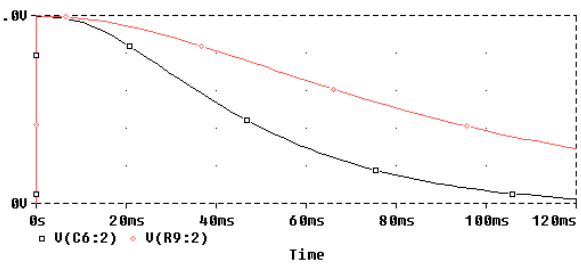

Figure 11 Comparison of time-domain behaviors of the 1956 Philbrick circuit (top curve without boxes) and the best circuit from generation 39 (bottom curve with boxes).

The legal criteria for obtaining a U. S. patent are that the proposed invention be "new" and "useful" and

> "... the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would [not] have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains." (35 *United States Code* 103a).

Since filing for a patent entails the expenditure of a considerable amount of time and money, patents are generally sought only if an individual or business believes the inventions are likely to be useful in the real world. Patents are only issued if an arms-length examiner is convinced that the proposed invention is novel, useful, and satisfies the statutory test for unobviousness.

The fact that genetic programming rediscovered both the topology and sizing of an electrical circuit that was unobvious "to a person having ordinary skill in the art" establishes that this evolved result satisfies Arthur Samuel's criterion (1983) for artificial intelligence

> "The aim [is] ... to get machines to exhibit behavior, which if done by humans, would be assumed to involve the use of intelligence."

7 The Illogical Nature of Creativity and Evolution

Many computer scientists and mathematicians unquestioningly assume that every problem-solving technique must be logically sound, deterministic, logically consistent, and parsimonious. Accordingly, most conventional methods of artificial intelligence and machine learning are constructed so as to possess these characteristics. However, logic does not govern two of the most important and significant types of processes for solving complex problems, namely the invention process (performed by creative humans) and the evolutionary process (occurring in nature).

A new idea that can be logically deduced from facts that are known in a field, using transformations that are known in a field, is not considered to be an invention. There must be what the patent law refers to as an "illogical step" (i.e., an unjustified step) to distinguish a putative invention from that which is readily deducible from that which is already known. Humans supply the critical ingredient of "illogic" to the invention process. Interestingly, everyday usage parallels the patent law concerning inventiveness: People who mechanically apply existing facts in well-known ways are summarily dismissed as being uncreative. Logical thinking is unquestionably useful for many purposes. It usually plays an important role in setting the stage for an invention. But, at the end of the day, logical thinking is not sufficient in the invention process.

Recalling his invention in 1927 of the negative feedback amplifier, Harold S. Black (1977) said,

> "Then came the morning of Tuesday, August 2, 1927, when the concept of the negative feedback amplifier came to me in a flash while I was crossing the Hudson River on the Lackawanna Ferry, on my way to work. For more than 50 years, I have pondered how and why the idea came, and I can't say any more today than I could that morning. All I know is that after several years of hard work on the problem, I suddenly realized that if I fed the amplifier output back to the input, in reverse phase, and kept the device from oscillating (singing, as we called it then), I would have exactly what I wanted: a means of canceling out the distortion of the output. I opened my morning newspaper and on a page of *The New York Times* I sketched a simple canonical diagram of a negative feedback amplifier plus the equations for the amplification with feedback."

Of course, inventors are not oblivious to logic and knowledge. They do not thrash around using blind random search. Black did not try to construct the negative feedback amplifier from neon bulbs or doorbells. Instead, "several years of hard work on the problem" set the stage and brought his thinking into the proximity of a solution. Then, at the critical moment, Black made his "illogical" leap. This unjustified leap constituted the invention.

The design of complex entities by the evolutionary process in nature is another important type of problem-solving that is not governed by logic. In nature, solutions to design problems are discovered by the probabilistic process of evolution and natural selection. This process is not guided by mathematical logic. Indeed, inconsistent and contradictory alternatives abound. In fact, such genetic diversity is necessary for the evolutionary process to succeed. Significantly, the solutions evolved by evolution and natural selection almost always differ from those created by conventional methods of artificial intelligence and machine learning in one very important respect. Evolved solutions are not brittle; they are usually able to grapple with the perpetual novelty of real environments.

8 Conclusion

Narrowly, we demonstrated the automatic synthesis, using genetic programming, of both the topology and sizing of a circuit composed of only resistors and capacitors that delivers a gain of greater than 2 and of a previously patented circuit suitable for preprocessing inputs to an oscilloscope. More broadly, we demonstrated how genetic programming can be can be used to automate the process of exploring queries, conjectures, and challenges concerning the existence of seemingly impossible entities. The approach described in the paper suggests a way in which genetic programming may be useful in automating the invention process.

References

Andre, David and Koza, John R. 1996. Parallel genetic programming: A scalable implementation using the transputer architecture. In Angeline, P. J. and Kinnear, K. E. Jr. (editors). 1996. *Advances in Genetic Programming 2*. Cambridge: MIT Press.

Angeline, Peter J. and Kinnear, Kenneth E. Jr. (editors). 1996. *Advances in Genetic Programming 2*. Cambridge, MA: The MIT Press.

Banzhaf, Wolfgang, Nordin, Peter, Keller, Robert E., and Francone, Frank D. 1998. *Genetic Programming – An Introduction*. San Francisco, CA: Morgan Kaufmann and Heidelberg: dpunkt.

Banzhaf, Wolfgang, Poli, Riccardo, Schoenauer, Marc, and Fogarty, Terence C. 1998. *Genetic Programming: First European Workshop. EuroGP'98. Paris, France, April 1998 Proceedings. Paris, France. April l998*. Lecture Notes in Computer Science. Volume 1391. Berlin, Germany: Springer-Verlag.

Bennett, Forrest H III, Koza, John R. Shipman, James, and Stiffelman, Oscar. 1999. Building a parallel computer system for $18,000 that performs a half peta-flop per day. Elsewhere in GECCO-99 proceedings.

Black, Harold S. 1977. Inventing the negative feedback amplifier. *IEEE Spectrum*. December 1977. Pp. 55 – 60.

Holland, John H. 1975. *Adaptation in Natural and Artificial Systems*. 2nd. Ed. Cambridge, MA: MIT Press.

Kinnear, Kenneth E. Jr. (editor). 1994. *Advances in Genetic Programming*. Cambridge, MA: MIT Press.

Koza, John R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: MIT Press.

Koza, John R. 1994a. *Genetic Programming II: Automatic Discovery of Reusable Programs.* Cambridge, MA: MIT Press.

Koza, John R. 1994b. *Genetic Programming II Videotape: The Next Generation*. Cambridge, MA: MIT Press.

Koza, John R. 1995. Evolving the architecture of a multi-part program in genetic programming using architecture-altering operations. In McDonnell, John R., Reynolds, Robert G., and Fogel, David B. (editors). *Evolutionary Programming IV: Proceedings of the Fourth Annual Conference on Evolutionary Programming*. Cambridge, MA: The MIT Press. Pages 695–717.

Koza, John R., Banzhaf, Wolfgang, Chellapilla, Kumar, Deb, Kalyanmoy, Dorigo, Marco, Fogel, David B., Garzon, Max H., Goldberg, David E., Iba, Hitoshi, and Riolo, Rick. (editors). 1998. *Genetic Programming 1998: Proceedings of the Third Annual Conference*. San Francisco, CA: Morgan Kaufmann.

Koza, John R., Bennett III, Forrest H, Andre, David, and Keane, Martin A. 1999a. *Genetic Programming III: Darwinian Invention and Problem Solving*. San Francisco, CA: Morgan Kaufmann. Forthcoming.

Koza, John R., Bennett III, Forrest H, Andre, David, and Keane, Martin A. 1999b. *Genetic Programming III Videotape*. San Francisco, CA: Morgan Kaufmann.

Koza, John R., Bennett III, Forrest H, Andre, David, Keane, Martin A, and Dunlap, Frank. 1997. Automated synthesis of analog electrical circuits by means of genetic programming. *IEEE Transactions on Evolutionary Computation*. 1(2). Pages 109 – 128.

Koza, John R., Deb, Kalyanmoy, Dorigo, Marco, Fogel, David B., Garzon, Max, Iba, Hitoshi, and Riolo, Rick L. (editors). 1997. *Genetic Programming 1997: Proceedings of the Second Annual Conference* San Francisco, CA: Morgan Kaufmann.

Koza, John R., Goldberg, David E., Fogel, David B., and Riolo, Rick L. (editors). 1996. *Genetic Programming 1996: Proceedings of the First Annual Conference*. Cambridge, MA: The MIT Press.

Koza, John R., and Rice, James P. 1992. *Genetic Programming: The Movie*. Cambridge, MA: MIT Press.

Langdon, W. B. 1998. *Genetic Programming and Data Structures: Genetic Programming + Data Structures = Automatic Programming!* Amsterdam: Kluwer.

Pease, Robert. 1996. What's all this R-C filter stuff, anyhow? *Electronic Design*. March 18, 1996.

Philbrick, George A. 1956. *Delayed Recovery Electric Filter Network*. Filed May 18, 1951. U. S. Patent 2,730,679. Issued January 10, 1956.

Poli, Riccardo, Nordin, Peter, Langdon, William B., and Fogarty, Terence C. 1999. *Genetic Programming: Second European Workshop. EuroGP'99. Proceedings*. Lecture Notes in Computer Science. Volume 1598. Berlin, Germany: Springer-Verlag.

Quarles, Thomas, Newton, A. R., Pederson, D. O., and Sangiovanni-Vincentelli, A. 1994. *SPICE 3 Version 3F5 User's Manual*. Department of Electrical Engineering and Computer Science, University of California. Berkeley, CA. March 1994.

Samuel, Arthur L. 1983. AI: Where it has been and where it is going. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*. Los Altos, CA: Morgan Kaufmann. Pages 1152 – 1157.

Spector, Lee, Langdon, William B., O'Reilly, Una-May, and Angeline, Peter (editors). 1999. *Advances in Genetic Programming 3*. Cambridge, MA: The MIT Press.

Sterling, Thomas L., Salmon, John, Becker, Donald J., and Savarese, Daniel F. 1999. *How to Build a Beowulf: A Guide to Implementation and Application of PC Clusters*. Cambridge, MA: MIT Press.

Zobel, Otto Julius. 1925. *Wave Filter*. Filed January 15, 1921. U. S. Patent 1,538,964. Issued May 26, 1925.