Mapping Real Numbers to Simple Resistor Networks

Samuel C. Hsieh and Sujan Pradhan

Abstract—A procedure to construct a simple resistor network for a given real number is presented. Given a rational number, the procedure is an algorithm that yields a finite simple network with the rational number as its equivalent resistance; given an irrational number, the procedure incrementally constructs a simple network by repeatedly adding unit resistors to the network such that the equivalent resistance of the network approaches the irrational number arbitrarily closely. The results of this procedure are confirmed computationally with Java code.

Index Terms—procedure, real numbers, resistor networks.

I. INTRODUCTION

A simple resistor network [1] is defined recursively to be either a unit resistor (i.e., a 1-ohm resistor), or a resistor network formed from a unit resistor connected in series or in parallel with a simple resistor network. For example, eight simple resistor networks are shown below (Fig. 1). All resistors used in this article are unit resistors.

![Networks](Network.png)

Fig. 1 Some Simple Resistor Networks

Network a in Fig. 1 is a single unit resistor and is the basis case of the recursive definition of simple resistor networks. Network b is derived from network a by connecting an extra unit resistor in parallel with the origin to lower the equivalent resistance of the network until one of the following cases occurs: a) the equivalent resistance becomes less than the given real number - in this case the equivalent resistance of the network approaches the irrational number as this construction process continues.

A one-to-one correspondence has been established in [1] between positive rational numbers and finite simple resistor networks. In this article we present a procedure to map positive real numbers to simple resistor networks. Each positive real number, which can be either rational or irrational, will be mapped to a simple resistor network, which may or may not be finite. Given a positive rational number, the procedure yields a finite simple network with the rational number as its resistance; given a positive irrational number, the procedure incrementally grows a simple network such that the resistance of the network gets closer and closer to the irrational number as the network grows, and the resistance of the network can get arbitrarily close to the given irrational number as this construction process continues.

II. A CONSTRUCTION PROCEDURE

The following procedure maps a real number to a simple resistor network. The input to the procedure is a positive real number x. The procedure constructs a simple resistor network N, which may or may not be finite depending on whether x is rational. The network N is initialized to be a single unit resistor, which is also the initial origin of the network. The origin of a network is the unit resistor to which an extra resistor will be added each time the network is enhanced. The notation eqres(N) denotes the equivalent resistance of a simple network N. For example, eqres(N) is 1 when N is the initial network – a single unit resistor.

```java
while ( eqres(N) is not x )
    if ( eqres(N) > x )
        replace the origin of N with the origin and an extra unit resistor in parallel
    else
        replace the origin of N with the origin and an extra unit resistor in series
//end if-else
//end while loop
```

Intuitively, when the resistance of the network is greater than the given real number, the procedure repeatedly adds unit resistors in parallel with the origin to lower the equivalent resistance of the network until one of the following cases occurs: a) the equivalent resistance becomes less than the given real number - in this case the equivalent

Manuscript received January 21, 2013.
S. C. Hsieh is with Computer Science Department, Ball State University, Muncie, IN 47306 USA (e-mail: shsieh@bsu.edu).
S. Pradhan is with Computer Science Department, Ball State University, Muncie, IN 47306 USA.
As shown in Table I, during the construction process, the resistance of the network fluctuates between lower and lower peaks and higher and higher valleys, and some of the intermediate networks, such as network c and network g, are neither peaks nor valleys.

For each network in Fig. 2, a net string is shown as bold-faced text. A net string is a concise text representation of a simple network. Every net string begins with the character 1, which denotes the origin. The letter P in a net string denotes a resistor in parallel and S denotes a resistor in series. For example, the net string for network a in Fig. 2 is 1. The net string for network b is 1P, which denotes the origin and a resistor in parallel and is derived by replacing the origin of network a (i.e., 1) with 1P - the origin and a resistor in parallel. The net string for network c is 1SP, which is derived by replacing the origin of network b with 1S - the origin and a resistor in series. The net strings of the other networks in Fig. 2 are derived similarly. Net strings will be used in the next two sections.

### III. CONVERGENCE AND LIMIT

Given a rational number, the construction procedure terminates, yielding a network with the given rational as the resistance of the network, since there is a one-to-one correspondence between positive rationals and finite simple resistor networks [1]. However, the resistance of a finite network is always a fraction, but an irrational number cannot be expressed as a fraction. We will now show that, when given an irrational, the procedure constructs a simple network with a resistance that can be arbitrarily close to the irrational.

First, we will show that the resistance of a network under construction for a given irrational must reach the first valley or first peak. If the irrational is less than 1, the first valley will be reached because repeatedly adding parallel resistors to the origin will eventually lower the resistance below the irrational; otherwise, the first peak will be reached because repeatedly adding series resistors will eventually raise the resistance above the irrational.

Next, we show that a peak is always followed by a valley and vice versa. Suppose the resistance of the network (represented by the net string) 1SX, where X is a string of S’s and P’s, is a peak. When the procedure adds n (>0) parallel resistors to the origin, the network becomes 1P_n…P_SX, where 1P_n…P_1 denotes n resistors in parallel with the origin. The resistance of 1P_n…P_1 falls as n rises, and the resistance approaches 0 as n approaches infinity. In other words, the resistance of 1P_n…P_1SX falls as n rises and the resistance approaches 1X as n approaches infinity. However, since the resistance of 1SX is a peak, the resistance of 1X must be less than the given irrational (because 1X is the network right before the last series resistor is added to reach the peak represented by 1SX). That is, there exists an integer k (>0) such that 1P_k…P_SX is a valley. That a valley is always followed by a peak can be similarly shown.

Next, we show that each peak is lower than the previous peak and each valley is higher than the previous valley. Let the resistance of the network 1SX, where X is a string of S’s

---

Table I. Construction of a Network with Resistance=0.72

<table>
<thead>
<tr>
<th>Network in Fig. 2</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resistance</td>
<td>1</td>
<td>1/2</td>
<td>2/3</td>
<td>3/4</td>
<td>5/7</td>
<td>8/11</td>
<td>13/18</td>
</tr>
<tr>
<td>Peak (P) or Valley (V)</td>
<td>V</td>
<td>P</td>
<td>V</td>
<td>P</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

As shown in Table I, during the construction process, the resistance of the network fluctuates between lower and lower peaks and higher and higher valleys, and some of the intermediate networks, such as network c and network g, are neither peaks nor valleys.
and P's, be the n th peak, where n > 0. Suppose j parallel resistors are added for the network resistance to reach a valley and then k series resistors are added to reach the (n+1) th peak, resulting in the network 1S X P j ..S P k SX. The resistance of 1S X ..S P j ..P k SX is smaller than 1 since it consists of j unit resistors P j ..P k connected in parallel with the network 1S X ..S P j ..P k SX. Hence, the resistance of 1S X ..P j ..S P k SX, which is the (n+1) th peak, must be lower than the resistance of 1SX, the n th peak. That each valley is higher than the previous valley can be similarly shown.

In summary, the resistance of a simple network under construction for a given irrational fluctuates between peaks and valleys as the peaks get lower and lower and the valleys get higher and higher. The peaks form a convergent sequence with the given irrational as the limit of the sequence, and so do the valleys.

IV. COMPUTATIONAL CONFIRMATION

The following Java code implements the construction procedure given previously:

```java
public static double eqres(String netString) {
    int den=1, num=1; //denominator and numerator
    int len=netString.length();
    for (int i=1; i<len; i++)
        if (netString.charAt(i) == 'P')
            den=den+num;
        else
            num=num+den;
    return (double) num / (double) den;
} //end eqres

Given a net string, the method eqres computes and returns the resistance of a network represented by the net string. Since the resistance of a finite simple network is always a rational, eqres computes the resistance as a fraction num/den. The method findNetwork constructs a network whose resistance value is the parameter real (within the given margin). The method reports the network as a net string and the equivalent resistance of the network. During the construction process, the method reports the peaks and valleys: when adding a series resistor to the origin causes the resistance of the network to rise above the given real, the resistance is a peak; when adding a parallel resistor to the origin causes the resistance of the network to fall below the given real, the resistance is a valley. As expected, the method call findNetwork(0.72, 0) (to construct a network with the resistance 0.72) outputs the following:

Net string = 1PPSPSSP
Resistance = 0.72

As another example, the method call findNetwork(Math.sqrt(2.0), 0) (to construct a network with the square root of 2 as its resistance) outputs the following:

Peak:   2.0
Valley: 1.3333333333333333
Peak:   1.4142857142857143
Valley: 1.411764705882353
Peak:   1.4146341463414633
Valley: 1.4142139267767408
Peak:   1.4142139988513232
Peak:   1.4142135731001355
Valley: 1.4142135605326258
Peak:   1.4142135626888697
Valley: 1.4142135623189167
Peak:   1.4142135623823906
Valley: 1.4142135623715002
Peak:   1.4142135623733687
Valley: 1.414213562373048
Peak:   1.4142135623731031
Valley: 1.4142135623730936
Peak:   1.4142135623730954

Netstring=1PPSSPSPPSSPPSSPPSSPSPPSSPPSSP
Resistance = 1.4142135623730951

It is worth noting that as the network is being constructed, the peaks and valleys converge towards the resistance Math.sqrt(2.0). Although the square root of 2 is an irrational number, the construction process terminates because in Java Math.sqrt(2.0) is the number 1.4142135623730951.

V. CONCLUSION

A procedure that maps real numbers to simple resistor networks is presented. Given a rational number, the procedure yields a finite simple network with the resistance equal to the given value. Given an irrational number, the procedure constructs a series of simple networks whose resistance values form a convergent sequence with the given
irrational as the limit of the sequence – that is, the procedure is able to construct a simple network with a resistance that is arbitrarily close to the irrational.

REFERENCES