Series-Parallel Circuit Construction: a Narrative with a Surprise.

Barrie Stokes

Discipline of Clinical Pharmacology
School of Medicine and Public Health
Faculty of Health
University of Newcastle
NSW Australia
ph (+61) 02 4921 1832
fax (+61) 02 4960 2088
Barrie.Stokes@newcastle.edu.au

■ Initialisation.

In[1]:=
```
Needs["DiscreteMath`Combinatorica`"]
Needs["Statistics`"]
Needs["Graphics`"]
```

■ Functions.

This function is just a slightly modified version of **Flatten[list,n]**, where the depth to which Flattening proceeds is specified by how close to the bottom of the nesting structure of "list" we wish to go. The idea will be to Flatten some lists which are complex Lists-of-Lists-of-Lists, etc., just to the required degree.

In[4]:=
```
Clear[ separateElements ]
separateElements[ list_List, d_ ] :=
 Flatten[ list, Length[ Dimensions[ list ] ] - d ]
```

```
In[6]:= Clear[series, paral]
        series[x_, y_] := x + y
        paral[x_, y_] := 1/(1/x + 1/y)
```

We need Boolean tests to say whether a character is or is not a left or a right parenthesis:

```
In[9]:= Clear[lbrQ]
        lbrQ[n_] := n == 40
```

```
In[11]:= Clear[nonLbrQ]
         nonLbrQ[n_] := n != 40
```

```
In[13]:= Clear[rbrQ]
         rbrQ[n_] := n == 41
```

```
In[15]:= Clear[nonRbrQ]
         nonRbrQ[n_] := n != 41
```

Mathematica already has a built-in Boolean test for a list:

```
In[17]:= ListQ /@ { aaa,  {aaa},  {{aaa}}  }
Out[17]= {False, True, True}
```

We need to define the complementary Boolean test for a non-list, to be used in the following function for "denesting" expressions like {{{a,b,c}}}:

```
In[18]:= Clear[nonListQ]
         nonListQ[x_] := !ListQ[x]
```

Here's what they do:

```
In[20]:= ListQ /@ { aaa,  {aaa},  {{aaa}}  }
         nonListQ /@ { aaa,  {aaa},  {{aaa}}  }
Out[20]= {False, True, True}
Out[21]= {True, False, False}
```

This function uses **nonListQ** to strip off repeated enclosing "{" and matching "}" braces from terms like {{{a,b,c}}}:

```
In[22]:= Clear[denestTerm]
         denestTerm[{a_?ListQ}] := denestTerm[a]
         (* remove extra sets of enclosing "{" and "}" until none remain *)
         denestTerm[a_] := a (* leave a non-nested list unaltered *)
         denestTerm[a_?nonListQ] := a (* leave isolated non-lists altered *)
```

These next two functions convert a nested representation to a circuit resistance expression via the T1 transformation rules in which "," means series and "},{" means parallel, and *vice versa* for the T2 rules.

In[26]:=
```
Clear[nestStructureToResistanceT1]
nestStructureToResistanceT1[originalTestExpression_] :=
 Module[{testExpression, codeExpression, series, paral, codeValue},

  (* convert originalTestExpression to a list of character codes *)
  testExpression = ToCharacterCode[ToString[originalTestExpression]];
  (* remove any spaces *)
  testExpression = testExpression //. {x___, 32, y___} → {x, y};
  (* convert all { to (, and all } to ) *)
  testExpression = testExpression //. {x___, 123, y___} → {x, 40, y};
  testExpression = testExpression //. {x___, 125, y___} → {x, 41, y};
  (* replace all "!),!(" with "~series~" *)
  testExpression =
   testExpression //. {u___, v_, w_?nonRbrQ, 44, x_?nonLbrQ, y_, z___} →
     {u, v, w, 126, 115, 101, 114, 105, 101, 115, 126, x, y, z};
  (* replace all "),!(" with "~series~" *)
  testExpression =
   testExpression //. {u___, v_, w_?rbrQ, 44, x_?nonLbrQ, y_, z___} →
     {u, v, w, 126, 115, 101, 114, 105, 101, 115, 126, x, y, z};
  (* replace all "!),(" with "~series~" *)
  testExpression =
   testExpression //. {u___, v_, w_?nonRbrQ, 44, x_?lbrQ, y_, z___} →
     {u, v, w, 126, 115, 101, 114, 105, 101, 115, 126, x, y, z};
  (* replace all "),(" with "~paral~" *)
  testExpression =
   testExpression //. {u___, v_, w_?rbrQ, 44, x_?lbrQ, y_, z___} →
     {u, v, w, 126, 112, 97, 114, 97, 108, 126, x, y, z};
  (* convert testExpression from list of character codes to a string,
   and then to code expression *)
 codeExpression = ToExpression[FromCharacterCode[testExpression]]
 ]
```

```
In[28]:= Clear[nestStructureToResistanceT2]
nestStructureToResistanceT2[originalTestExpression_] :=
 Module[{testExpression, codeExpression, series, paral, codeValue},
  (* convert originalTestExpression to a list of character codes *)
  testExpression = ToCharacterCode[ToString[originalTestExpression]];
  (* remove any spaces *)
  testExpression = testExpression //. {x___, 32, y___} → {x, y} ;
  (* remove enclosing { and } *)
  (*** testExpression = Drop[ Rest[ testExpression ], -1 ]; ***)
  (* convert all { to (, and all } to ) *)
  testExpression = testExpression //. {x___, 123, y___} → {x, 40, y};
  testExpression = testExpression //. {x___, 125, y___} → {x, 41, y};
  (* replace all "!),!(" with "~paral~" *)
  testExpression =
   testExpression //. {u___, v_, w_?nonRbrQ, 44, x_?nonLbrQ, y_, z___} →
     {u, v, w, 126, 112, 97, 114, 97, 108, 126, x, y, z} ;
  (* replace all "),!(" with "~paral~" *)
  testExpression =
   testExpression //. {u___, v_, w_?rbrQ, 44, x_?nonLbrQ, y_, z___} →
     {u, v, w, 126, 112, 97, 114, 97, 108, 126, x, y, z} ;
  (* replace all "!),(" with "~paral~" *)
  testExpression =
   testExpression //. {u___, v_, w_?nonRbrQ, 44, x_?lbrQ, y_, z___} →
     {u, v, w, 126, 112, 97, 114, 97, 108, 126, x, y, z} ;
  (* replace all "),(" with "~series~" *)
  testExpression =
   testExpression //. {u___, v_, w_?rbrQ, 44, x_?lbrQ, y_, z___} →
     {u, v, w, 126, 115, 101, 114, 105, 101, 115, 126, x, y, z} ;
  (* convert testExpression from list of character codes to a string,
   and then to code expression *)
 (*** Print[ originalTestExpression   ]; ***)
  (*** Print[ FromCharacterCode[ testExpression ]  ]; ***)
 codeExpression = ToExpression[FromCharacterCode[testExpression]]
  ]
```

Note: Although this code looks fussy, using replacement rules on lists of character codes, here is an estimate of the average timing for one conversion of a 7-element nested expression on a Toshiba TE2000 1.7 MHz Pentium III laptop.

We also illustrate the complementarity of the essentially series and essentially parallel circuits represented by the same nested structure.

In[30]:=
```
nestStructureToResistanceT1[ {{t}, {u, {{v, {{w}, {x}}, y}, {z}}}} ]
nestStructureToResistanceT2[ {{t}, {u, {{v, {{w}, {x}}, y}, {z}}}} ]
Do[ nestStructureToResistanceT1[ {{t}, {u, {{v, {{w}, {x}}, y}, {z}}}} ],
   {10000} ] // Timing;
%[[1]] / 10000
1
%
```

Out[30]=
$$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v+\frac{1}{\frac{1}{w}+\frac{1}{x}}}+y}+\frac{1}{z}}}$$

Out[31]=
$$t+\frac{1}{\frac{1}{u}+\frac{1}{\frac{1}{\frac{1}{v}+\frac{1}{w+\frac{1}{x}+\frac{1}{y}}}+z}}$$

Out[33]= `0.0045025 Second`

Out[34]=
$$\frac{222.099}{\text{Second}}$$

For use in certain of the following defined functions we need a couple of custom Boolean functions. The first is …

In[35]:=
```
Clear[ onlyVariableQ ]
onlyVariableQ[ x__ ] :=
 ! ((x === Plus ) || (x === Power) || (x === -1) || (Length[x] > 0))
```

We also define the logically complementary function, for explanatory reasons.

In[37]:=
```
Clear[ notOnlyVariableQ ]
notOnlyVariableQ[ x__ ] := (x === Plus ) || (x === Power) || (x === -1)
```

Here's how they work:

In[39]:=
```
onlyVariableQ /@    {x1, x12, Power, Plus, -1, 1/x1}
notOnlyVariableQ /@ {x1, x12, Power, Plus, -1, 1/x1}
```

Out[39]= `{True, True, False, False, False, False}`

Out[40]= `{False, False, True, True, True, False}`

The following applications of **`notOnlyVariableQ[]`** shows that in the structure expressions of two topologically identical, i.e., equivalent circuits, the Heads Power and Plus, and the -1 involved in a reciprocal, can be in different places, so this turns out to be no basis for an equivalence test.

In[41]:=
```
Position[ 1/(1/x1 + 1/(1/(1/x2 + 1/x3) + x4)), _?notOnlyVariableQ]
```

Out[41]=
```
{{0}, {1, 0}, {1, 1, 0}, {1, 1, 2}, {1, 2, 0}, {1, 2, 1, 0}, {1, 2, 1, 1, 0},
 {1, 2, 1, 1, 1, 0}, {1, 2, 1, 1, 1, 1, 0}, {1, 2, 1, 1, 1, 1, 2},
 {1, 2, 1, 1, 1, 2, 0}, {1, 2, 1, 1, 1, 2, 2}, {1, 2, 1, 1, 2}, {1, 2, 2}, {2}}
```

In[42]:=
```
Position[ 1/(1/x1 + 1/(x2 + 1/(1/x3 + 1/x4))), _?notOnlyVariableQ]
```

Out[42]=
```
{{0}, {1, 0}, {1, 1, 0}, {1, 1, 2}, {1, 2, 0}, {1, 2, 1, 0}, {1, 2, 1, 2, 0},
 {1, 2, 1, 2, 1, 0}, {1, 2, 1, 2, 1, 1, 0}, {1, 2, 1, 2, 1, 1, 2},
 {1, 2, 1, 2, 1, 2, 0}, {1, 2, 1, 2, 1, 2, 2}, {1, 2, 1, 2, 2}, {1, 2, 2}, {2}}
```

The complementary Boolean test **`onlyVariableQ[]`** will turn out to be useful …

In[43]:=
```
Position[ 1/(1/x1 + 1/(1/(1/x2 + 1/x3) + x4)), _?onlyVariableQ]
```

Out[43]=
```
{{1, 1, 1}, {1, 2, 1, 1, 1, 1, 1}, {1, 2, 1, 1, 1, 2, 1}, {1, 2, 1, 2}}
```

In[44]:=
```
Position[ 1/(1/x1 + 1/(x2 + 1/(1/x3 + 1/x4))), _?onlyVariableQ]
```

Out[44]=
```
{{1, 1, 1}, {1, 2, 1, 1}, {1, 2, 1, 2, 1, 1, 1}, {1, 2, 1, 2, 1, 2, 1}}
```

… because it has clearly found the positions of the different symbols x1, x2, etc., and in combination with the built-in function **`Extract[]`**, we can reproduces a list of all the symbols in a structure expression for a circuit.

In[45]:=
```
Clear[ extractElements ]
extractElements[ struct_ ] :=
 Extract[ struct,  Position[ struct, _?onlyVariableQ] ]
```

A test …

In[47]:=
```
extractElements[ 1/(1/x3 + 1/(x2 + 1/(1/x1 + 1/(x4 + x5 + 1/x6)))) ]
```

Out[47]=
```
{x3, x2, x1, x4, x5, x6}
```

`extractElements[]` is used inside the following function, since we need a list of the symbols in each of two structures we are testing for equivalence in order to construct the rules for substituting different highly precise numerical values into the two structures in all possible ways.

In[48]:=
```
equivalentStructuresQ[ struct1_, struct2_ ] :=
 Module[
  {perms, nPerms, elementSymbols1, elementSymbols2, nElements,
   allRules1, allRules2, allCircuitValues1, allCircuitValues2},

  elementSymbols1 = Sort[ extractElements[ struct1 ] ];
  elementSymbols2 = Sort[ extractElements[ struct2 ] ];

  If[ Length[ elementSymbols1 ] == Length[ elementSymbols2 ],
   (* circuits must at least have same number of symbols *)
   nElements = Length[ elementSymbols1 ];
   (perms = Permutations[ Table[ Random[], {nElements} ] ];
 nPerms = Length[ perms ];
 allRules1 = {};
 Do[ AppendTo[ allRules1,
       Thread[ Rule[ elementSymbols1, perms[[i]] ] ] ], {i, 1, nPerms} ];
  allRules2 = {};
 Do[ AppendTo[ allRules2,
       Thread[ Rule[ elementSymbols2, perms[[i]] ] ] ], {i, 1, nPerms} ];
  allCircuitValues1 = Release[ Hold[ struct1 ] /. allRules1 ];
  (* all possible substitutions in structure 1 *)
  allCircuitValues2 = Release[ Hold[ struct2 ] /. allRules2 ];
  (* all possible substitutions in structure 2 *)
  Length[ Intersection[ allCircuitValues1, allCircuitValues2 ] ] != 0),
 (* any common values? *)
 False (* structures don't have same symbols *)
 ]
]
```

Topologically identical, i.e., equivalent, circuits with same symbols:

In[49]:=
$$\texttt{equivalentStructuresQ}\left[\frac{1}{\frac{1}{x1+x2}+\frac{1}{x3}+\frac{1}{x4}},\ \frac{1}{\frac{1}{x1}+\frac{1}{x2+x3}+\frac{1}{x4}}\right]$$

Out[49]= `True`

Topologically identical, i.e., equivalent, circuits with different symbols:

In[50]:=
$$\texttt{equivalentStructuresQ}\left[\frac{1}{\frac{1}{aaa+bbb}+\frac{1}{ccc}+\frac{1}{ddd}},\ \frac{1}{\frac{1}{x1}+\frac{1}{x2+x3}+\frac{1}{x4}}\right]$$

Out[50]= `True`

Topologically non-identical, i.e., non-equivalent, circuits with same symbols:

In[51]:=
$$\texttt{equivalentStructuresQ}\left[\frac{1}{\frac{1}{x1+x2}+\frac{1}{x3}+\frac{1}{x4}},\ x1+\frac{1}{\frac{1}{x2}+\frac{1}{x3}+\frac{1}{x4}}\right]$$

Out[51]= `False`

Topologically non-identical, i.e., non-equivalent, circuits with same symbols:

In[52]:=
$$\texttt{equivalentStructuresQ}\left[\frac{1}{\frac{1}{x1}+\frac{1}{x2+\frac{1}{\frac{1}{x3}+\frac{1}{x4}}}},\ x1+\frac{1}{\frac{1}{x2}+\frac{1}{x3}+\frac{1}{x4}}\right]$$

Out[52]= `False`

Topologically identical, i.e., equivalent, circuits with same symbols:

In[53]:=
$$\texttt{equivalentStructuresQ}\left[\frac{1}{\frac{1}{x1}+\frac{1}{x2+\frac{1}{\frac{1}{x3}+\frac{1}{x4}}}},\ \frac{1}{\frac{1}{x1+\frac{1}{\frac{1}{x2}+\frac{1}{x3}}}+\frac{1}{x4}}\right]$$

Out[53]= `True`

Topologically non-identical, i.e., nonequivalent, circuits because of differing numbers of elements:

In[54]:=
$$\texttt{equivalentStructuresQ}\left[\frac{1}{\frac{1}{x1}+\frac{1}{x2+\frac{1}{\frac{1}{x3}+\frac{1}{x4}}}},\ \frac{1}{x1+\frac{1}{\frac{1}{x2}+\frac{1}{x3}}}\right]$$

Out[54]= `False`

This function "repairs" the effect of **`Sort[]`** applied to individual circuit expressions when they represent an essentially parallel circuit.

In[55]:=
```
Clear[repairFunction]

repairFunction[x__] := If[Head[x] === Plus, x, 1/x[[2]]]
```

■ Section 1 - Introduction.

The modern digital systems—CD, HDCD, DVD-A, SACD—for sound reproduction were preceded by an electromechanical system which was initially developed before 1900, and continues to be refined in the present day. The flat black gramophone record, or LP (for Long Playing) was introduced in the mid 1950s, and audiophiles regard the period from then until around 1970 as the "Golden Age of the LP".

It is possible today to purchase extremely expensive turntables, arms, and cartridges to play both old and newly pressed LPs, and many audiophiles in what is termed the high end of the market seriously claim that such equipment produces music of even greater fidelity than the best of modern digital systems.

The basic phonographic mechanism for storing and recovering the oscillating voltage signal originally produced by the microphone in the original recording venue is illustrated in Figures 1 and 3.

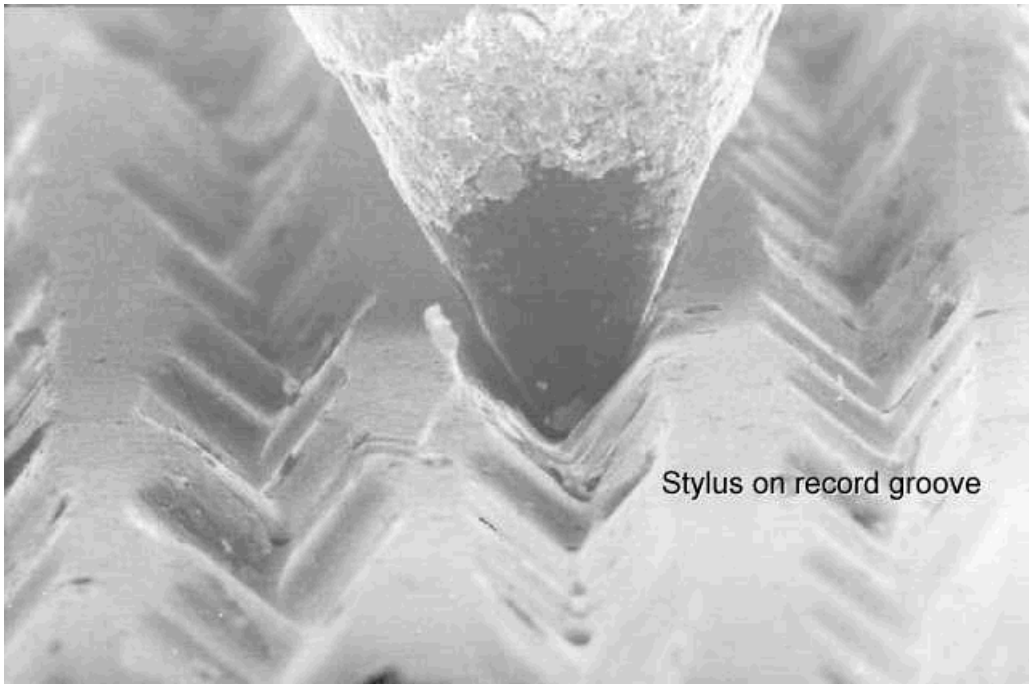


Figure 1. The tip of the stylus rests in the wiggly groove, and the rotation of the record means that the stylus is made to oscillate both laterally and vertically as it "tracks" the groove. Typical groove width is 75 microns (75×10^{-6} m). Typical dimensions of the tip of the stylus, which generally has an elliptical cross-section, are 5-6 microns by 35-120 microns. The wavelength of a 20 kHz signal near the inner grooves of a 30 cm LP is around 18 microns, while a medium level 1 kHz signal has a sideways amplitude of about 11 microns. The same signal at a still-audible 40 dB lower level has an amplitude of around 110×10^{-9} m, which is less than the wavelength of visible light. A very soft 10 kHz signal requires an amplitude of 5×10^{-9} m. Modern cartridges can trace such a waveform, and the signal when amplified can easily be heard as part of a musical presentation. The power output of a moving coil cartridge at 1 kHz is around 15×10^{-9} watts. (Image taken by Roger Heady, ANUEMU ★)

The stylus is attached to a light but rigid cantilever, a small thin metal tube, often beryllium or boron/aluminium, which is suspended in a suitable elastomeric—butyl rubber or neoprene—mounting. This arrangement allows the oscillations of the stylus to be faithfully transmitted to the end of the cantilever carrying either a magnet or a coil. Faraday's law says that if an electrically conductive loop—here the turns of the coil— is placed in a time-varying magnetic field,

the varying magnetic flux through the loop induces an emf ("electro-motive force") in the loop which sets electrons in motion, i.e., produces a current flow in the loop. As a physicist would say,

$$\oint \mathbf{E}.d\mathbf{L} = \partial_t \Phi_B,$$

where **E** represents the induced electric field which is tangential to the coil and causes a small electrical current to flow, and $\partial_t \Phi_B$ represents the rate of change of the magnetic flux through the coil. Faraday's law equally describes the complementary situation in which the coil is attached to the cantilever, and fixed magnets are positioned along the axis of the coil, the moving *coil* (MC) design.

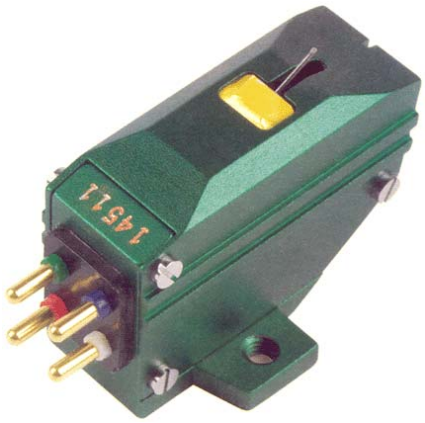


Figure 2. A modern gramophone cartridge, showing the cantilever carrying the diamond stylus, and the 4 connecting pins for the two stereo signals.

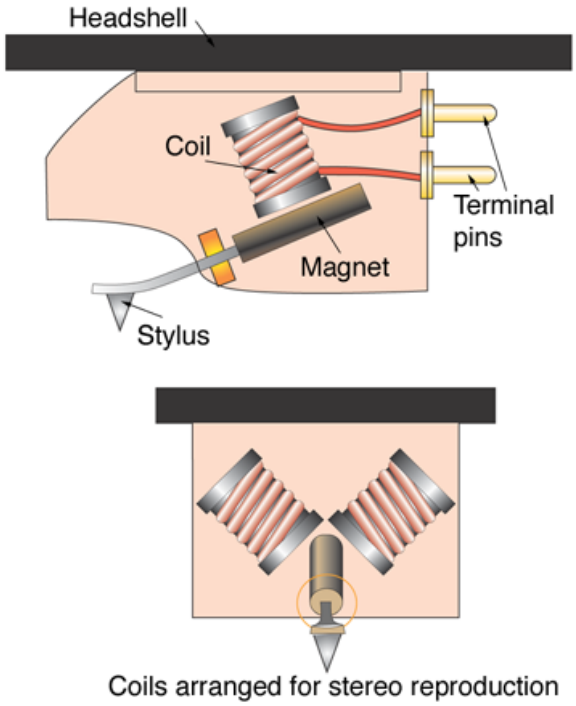


Figure 3. A highly schematic diagram of the basic mechanism of a phono cartridge, showing the stylus and cantilever, the pliable mounting block (orange), and in this case the magnet which moves relative to the coil. The front-on diagram shows the 45° arrangement of separate magnet/coils for detecting separate left and right stereo signals.The movements of the stylus cause the magnets to oscillate along the axes of the coils, inducing corresponding oscillating currents to flow in the coils (Image:Mag Cartridge.png ✱).

Because the physical amplitude of the oscillations is small, as are the magnets and the coil, the voltage signal which appears at the terminals pins on the back of the cartridge body (Figures 2 and 3) is of the order of a few millivolts in the case of an MM cartridge, and less than one millivolt for an MC cartridge. The signal is therefore passed first to a low-noise high quality amplifying circuit which boosts the voltage levels to those produced by other common input sources such as CD players, radio tuners, cassette decks, and so on. This circuit is known as the phono preamplifier, coming as it does before control unit and the further power amplification required to eventually produce sound from a loudspeaker.

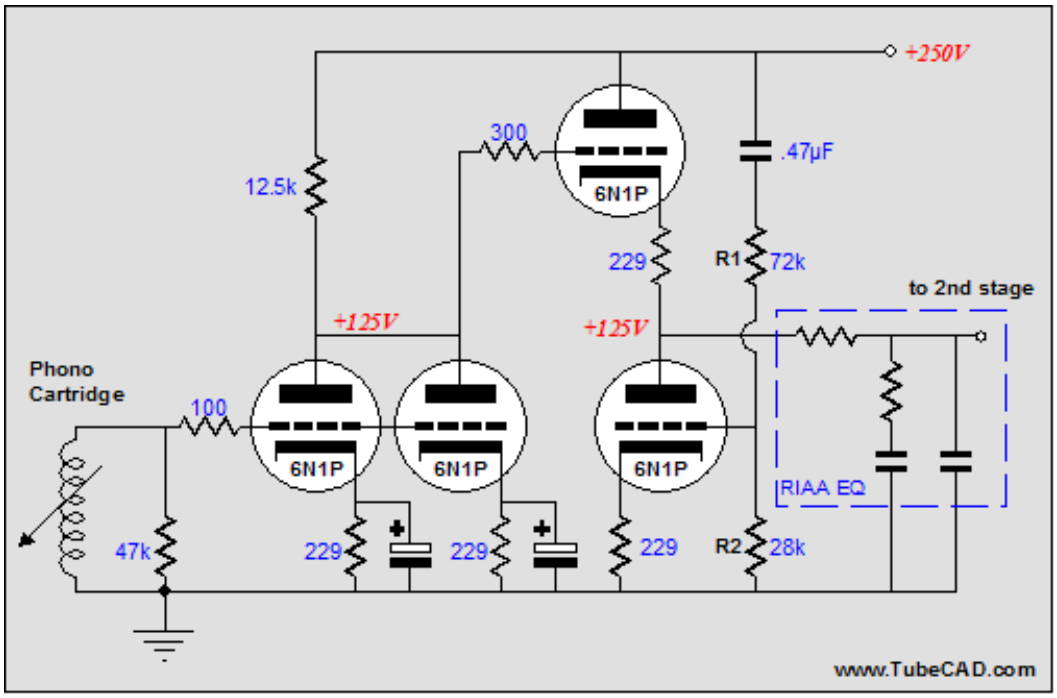


Figure 4. An example of a valve phono preamplifier circuit (tube phone preamplifier ★).

Figure 4 shows a valve phono preamplifier circuit, a critical part of which is the so-called load resistor (the -/\/\/\- symbol) marked here as being 47k, i.e. 47,000 ohms.

It turns out that the ohmic impedance (resistance) of the load of the cartridge has a non-trivial effect on the ultimate fidelity of the sound produced by the complete sound reproduction system.

Several high fidelity equipment manufacturers supply a selection of high-precision resistors intended to be used by the purchaser to form a small resistive network having a total resistance giving optimal sound performance.

Figure 5. A typical modern phono preamplifier, in this case a valve or tube design (PH3 / PH3SE Stereo Phono Preamplifier ✸).

In 1990 the I purchased such a device, shown in Figure 5. We quote from the manufacturers website:

> "The PH3 is a phono preamplification stage intended for use with an active line-level stereo preamplifier. Fully single-ended from input to output, the PH3 uses a combination of J-FET input stage technology (for low inherent circuit noise) and class A operation vacuum-tube gain via three 6922 twin triodes. Total gain of the PH3 is 54 dB (non-adjustable). Used with typical line preamplifiers having 12 to 18 dB of gain, the PH3 is suitable for phono cartridges having .25 mV output or higher. Input impedance is factory-set at 47K ohms, and input capacitance at 90 pF. **For optimal matching to moving-coil and moving-magnet phono cartridges, both input impedance and input capacitance may be adjusted via simple resistor and capacitor changes.**"

Figure 6. Some typical resistors as used in discrete component, as opposed to highly integrated, electronics often found in high fidelity equipment (carbon film resistors ✸).

The problem we address below is obtaining an explicit representation of all of the possible series-parallel circuits that can be constructed using a given set of resistors. A series arrangement is shown in Figure 7, a parallel in Figure 8, and Figure 9 shows the simplest series-parallel circuit.

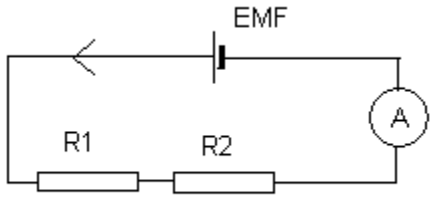


Figure 7. R1 and R2 are said to be in series (Series Electrical Circuit ✱).

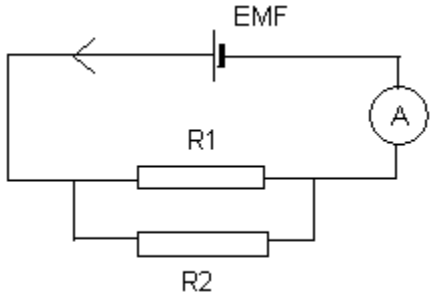


Figure 8. R1 and R2 are said to be in parallel (Parallel Electrical Circuit ✱).

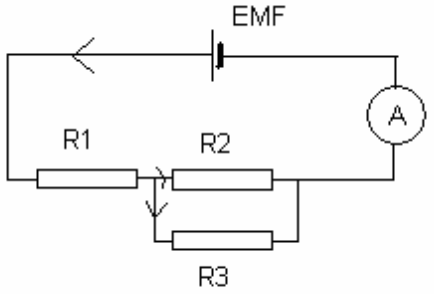


Figure 9. R1, R2, and R3 are said to be in a series-parallel arrangement, represented by X_1 (X_2 + X_3) in the left column of the table in Figure 10 below (Series-Parallel Electrical Circuit ✱).

■ Section 2 - Mathematical Background.

The earliest reference to this problem is Mac Mahon [1 ✱] in 1892, and luminaries such as Claude Shannon of information theory fame published on this problem in 1942 [2 ✱]. We have found the papers by Golinelli [3 ✱] and Lomnicki [4 ✱] very helpful in arriving at a *Mathematica* representation for a series-parallel arrangement. Amengual [5 ✱], Sloane [6 ✱], and Weisstein [8 ✱]are further references.

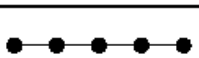
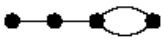
number of elements	essentially series		essentially parallel		number of networks
1		X_1		X_1	1
2		X_1X_2		$X_1 + X_2$	2
3		$X_1X_2X_3$		$X_1 + X_2 + X_3$	4
		$X_1(X_2 + X_3)$		$X_1 + X_2X_3$	
4		$X_1X_2X_3X_4$		$X_1 + X_2 + X_3 + X_4$	10
		$X_1X_2(X_3 + X_4)$		$X_1 + X_2 + X_3X_4$	
		$X_1(X_2 + X_3 + X_4)$		$X_1 + X_2X_3X_4$	
		$X_1(X_2 + X_3X_4)$		$X_1 + X_2(X_3 + X_4)$	
		$(X_1 + X_2)(X_3 + X_4)$		$X_1X_2 + X_3X_4$	

Figure 10. This is Table 1 from Golinelli [3 ★], showing the actual possible circuits for 1, 2, 3, and 4 resistances, or more generally circuit elements, and the corresponding notation.

The notation for series-parallel networks used in Figure 10 derives directly from the Boolean notation in which X.Y (the *And* operator) represents X and Y, corresponding to the idea that current flowing through X and Y in series flows through both X *and* Y, while for elements in parallel the current can flow through X *or* Y, and the representation X+Y uses the Boolean *Or* operator.
For our present purpose we consider the representations $(X_1+X_2).X_3 + X_4$ and $X_1 + X_2.(X_3+X_4)$ to be equivalent since they differ only by permutation of the variables.
This representation can be easily read off as the corresponding circuit, so an analogous representation in *Mathematica* has been devised, based on the braces-based notation{a, b, c, …} used for lists.

In our representation scheme 4 resistors in series—a.b.c.d—are represented as

```
{a, b, c, d}
```

and 4 in parallel—a+b+c+d—as

```
{{a}, {b}, {c}, {d}}
```

The Boolean *And* operator is thus represented by simple adjacency within a list, i.e., by the comma "," between the elements, while the Boolean *Or* is represented by the element(s) in parallel appearing in adjacent lists, i.e., by "},{" between the paralleled elements. We refer to a such list structure as the "nesting" structure for the corresponding circuit.

The formulae for the total resistance of a number of resistors in series and in parallel are well known to beginning physics student, electronics experts, and mathematicians of various stripes. For a series configuration such as the circuit represented by $X_1.X_2.X_3$ in Golinelli's Figure 10 it is …

$$X_{\text{total}} = X_1 + X_2 + X_3$$

… while for a parallel configuration such as is represented by $X_1 + X_2 + X_3$, the formula is the elegant

$$X_{\text{total}} = \frac{1}{\frac{1}{X_1} + \frac{1}{X_2} + \frac{1}{X_3}}.$$

We will convert generated nesting representations to series-parallel expressions via a function such as **`nestStructureToResistanceT1[]`**:

```
In[57]:= nestStructureToResistanceT1[ {a, b, c, d} ]
Out[57]= a + b + c + d
```

In[58]:=
```
nestStructureToResistanceT1[ {{a}, {b}, {c}, {d}} ]
```

Out[58]=
$$\frac{1}{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}+\frac{1}{d}}$$

While it is true that this last expression for parallel resistors can be more conventionally written as

In[59]:=
```
1/(1/x1 + 1/x2 + 1/x3) // Together
```

Out[59]=
$$\frac{x_1\, x_2\, x_3}{x_1\, x_2 + x_1\, x_3 + x_2\, x_3}$$

it will prove to be much more convenient to keep to the "harmonic mean" form when we want to interpret easily the representation of more complex circuits, and *Mathematica* allows us to do this throughout.

In the following explanation of the method we consider for illustration the case of 4 circuit elements.

The problem is to generate all of the "nested" representations of the 10 possible distinct circuits consisting of 4 elements.

As the layout of the Table in Figure 10 clearly suggests, for every set of resistors there are an equal number of circuits termed *essentially series* and *essentially parallel*, and so in order not to miss any possibilities, we have coded two complementary functions, **`nestStructureToResistanceT1[]`** and **`nestStructureToResistanceT2[]`**, the second of which does what you would expect, …

In[60]:=
```
nestStructureToResistanceT2[ {a, b, c, d} ]
```

Out[60]=
$$\frac{1}{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}+\frac{1}{d}}$$

In[61]:=
```
nestStructureToResistanceT2[ {{a}, {b}, {c}, {d}} ]
```

Out[61]=
$$a + b + c + d$$

These transformations will be required at the end of the process of generating all the necessary nested representations, so that we can easily recognise the various series-parallel circuits represented.

■ Section 3 - Series-Parallel Circuits with 4 Element: Generating all the Nested Representations.

We start by generating a set of "basic" lists, each having 4 members, and in which all the possible combinations of braced—{a}—and unbraced—a—members occur. **`Outer[]`** is powerful built-in Mathematica function which gets things started. The need for the **`symbolSetAlternate`** will be explained below.

In[62]:=
```
symbolSet = {t, u, v, w}; Clear[ t, u, v, w ]
symbolSetAlternate = {a, b, c, d}; Clear[ a, b, c, d ]
```

In[64]:=
```
startList = Map[ ({#, {#}}) &, symbolSet ];
basicLists = Outer[List, Sequence @@ startList, 1 ]
Dimensions @ %
```

Out[65]=
```
{{{{{t, u, v, w}, {t, u, v, {w}}}, {{t, u, {v}, w}, {t, u, {v}, {w}}}},
  {{{t, {u}, v, w}, {t, {u}, v, {w}}},
   {{t, {u}, {v}, w}, {t, {u}, {v}, {w}}}},
 {{{{t}, u, v, w}, {{t}, u, v, {w}}}, {{{t}, u, {v}, w},
    {{t}, u, {v}, {w}}}, {{{t}, {u}, v, w}, {{t}, {u}, v, {w}}},
  {{{t}, {u}, {v}, w}, {{t}, {u}, {v}, {w}}}}}
```

Out[66]=
```
{2, 2, 2, 2, 4}
```

The Dimension command tells us that we have lists with 4 members inside the highly nested result from **Outer[]**]. We bring them to the surface with …

```
In[67]:= singleNestedOrNot = separateElements[ basicLists, 2 ]
         Dimensions @ %
```

```
Out[67]= {{t, u, v, w}, {t, u, v, {w}}, {t, u, {v}, w}, {t, u, {v}, {w}}, {t, {u}, v, w},
          {t, {u}, v, {w}}, {t, {u}, {v}, w}, {t, {u}, {v}, {w}}, {{t}, u, v, w},
          {{t}, u, v, {w}}, {{t}, u, {v}, w}, {{t}, u, {v}, {w}}, {{t}, {u}, v, w},
          {{t}, {u}, v, {w}}, {{t}, {u}, {v}, w}, {{t}, {u}, {v}, {w}}}

Out[68]= {16, 4}
```

… and assign this initial list of lists, in which each symbol is either in braces "{a}" or not, to a variable denoting that this is the starting point of an iterative process (keeping a close eye on the Dimensions of all of our results):

```
In[69]:= list0 = singleNestedOrNot
         Dimensions @ %
```

```
Out[69]= {{t, u, v, w}, {t, u, v, {w}}, {t, u, {v}, w}, {t, u, {v}, {w}}, {t, {u}, v, w},
          {t, {u}, v, {w}}, {t, {u}, {v}, w}, {t, {u}, {v}, {w}}, {{t}, u, v, w},
          {{t}, u, v, {w}}, {{t}, u, {v}, w}, {{t}, u, {v}, {w}}, {{t}, {u}, v, w},
          {{t}, {u}, v, {w}}, {{t}, {u}, {v}, w}, {{t}, {u}, {v}, {w}}}

Out[70]= {16, 4}
```

Next we use the powerful *Mathematica* command **ReplaceList[ ]** to apply in all possible ways a rule which gives lots of expressions with the next level of nesting. Recall that x___ is a pattern object that can stand for any sequence of zero or more *Mathematica* expressions, and that x__ is a pattern object that can stand for any sequence of one or more *Mathematica* expressions.

```
In[71]:= list1 =
           Map[ (ReplaceList[ #, {{a___, b__, c___} → {a, {b}, c}} ]) &, list0 ];
         list1 // Short
```

```
Out[72]//Short= { <<1>> }
```

Because there are 10 ways to apply the rule {{a___,b__,c___}→{a,{b},c}} to a list of 4 terms (4 + 3 + 2 + 1)…

```
In[73]:= ReplaceList[ {aa, bb, cc, dd}, {{a___, b__, c___} → {a, {b}, c}} ]
         Dimensions @ %
```

```
Out[73]= {{{aa}, bb, cc, dd}, {{aa, bb}, cc, dd},
          {aa, {bb}, cc, dd}, {{aa, bb, cc}, dd},
          {aa, {bb, cc}, dd}, {aa, bb, {cc}, dd}, {{aa, bb, cc, dd}},
          {aa, {bb, cc, dd}}, {aa, bb, {cc, dd}}, {aa, bb, cc, {dd}}}

Out[74]= {10}
```

… and we started with a list of 2^4= 16 basic lists, list1 should have Dimensions {16, 10}:

```
In[75]:= Dimensions @ list1
```

```
Out[75]= {16, 10}
```

By judicious choice of the extent of Flattening—the "d" parameter of **separateElements[list,d]** is set to 1—we reveal the members of list1 each having 4 symbols but with a great variety of internal nesting, applying the incredibly useful *Mathematica* function **Union[]** to remove any repeated members :

```
In[76]:= list1 = separateElements[ list1, 1 ] // Union;
         list1 // Short
         Dimensions @ %

Out[77]//Short= {{{t, u, v, w}}, {{t, u, v, {w}}}, {{t, u, {v}, w}}, {{t, u, {v}, {w}}},
                <<136>>, {{{t}}, {u}, v, {w}}, {{{t}}, {u}, {v}, w}, {{{t}}, {u}, {v}, {w}}}

Out[78]= {143}
```

The variety of internal nesting in these representations can be shown with …

```
In[79]:= Depth /@ list1

Out[79]= {3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 4, 4, 4, 4,
          4, 3, 4, 4, 4, 4, 4, 4, 4, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
          4, 3, 4, 4, 4, 3, 4, 4, 4, 3, 3, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 4, 3, 4, 4,
          4, 3, 3, 4, 4, 4, 4, 4, 4, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
          4, 3, 4, 3, 3, 4, 4, 4, 3, 3, 4, 3, 3, 4, 4, 4, 4, 4, 4, 4, 3, 3, 4, 3,
          3, 4, 4, 4, 3, 3, 4, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4}
```

In the next iteration we use **ReplaceList[]** again to apply the same rule as before …

```
In[80]:= list2 =
           Map[ (ReplaceList[ #, {{a___, b__, c___} → {a, {b}, c}} ]) &, list1 ];
         list2 // Short
         Dimensions @ %

Out[81]//Short= {{{{{t, u, v, w}}}}, <<141>>,
                 {{{{{t}}}}, {u}, {v}, {w}}, {{{{t}}}, {u}}, {v}, {w}},
                  <<6>>, {{{t}}, {u}, {{v}, {w}}}, {{{t}}, {u}, {v}, {{w}}}}}

Out[82]= {143}
```

… and as before we separate out the members we want. The "d" parameter of **separateElements[list,d]** is now set to 0 …

```
In[83]:= list2 = separateElements[ list2, 0 ] // Union;
         list2 // Short
         Dimensions @ %

Out[84]//Short= {{{{t, u, v, w}}}, {{{t, u, v, {w}}}}, {{{t, u, {v}, w}}}, <<631>>,
                 {{{{t}}}, {u}, v, {w}}, {{{{t}}}, {u}, {v}, w}, {{{{t}}}, {u}, {v}, {w}}}

Out[85]= {637}
```

If we look at the members of $\texttt{list}_2$ we see that there is a lot of superfluous outside nesting:

```
In[86]:= list2 [[15]]

Out[86]= {{{{t}, {u}, {v}, w}}}
```

According to our interpretation of nestings, {{y}} is not different from {y}. The function **denestTerm[]** removes these extra braces:

```
In[87]:= denestTerm[ aaa ]

Out[87]= aaa

In[88]:= denestTerm[ {{{{{{{{a, b, c}}}}}}}} ]

Out[88]= {a, b, c}

In[89]:= denestTerm[ list2 [[15]] ]

Out[89]= {{t}, {u}, {v}, w}
```

At this point we apply our denesting function to each of these expressions, to remove superfluous bracing:

```
In[90]:= list2Denested = Map[ (denestTerm[ # ]) &, list2 , {1, ∞} ];
         list2Denested // Short
         Dimensions @ %
```

```
Out[91]//Short= {{t, u, v, w}, {t, u, v, {w}}, {t, u, {v}, w},
                 {t, u, {v}, {w}}, <<629>>, {{t}, {u}, v, w},
                 {{t}, {u}, v, {w}}, {{t}, {u}, {v}, w}, {{t}, {u}, {v}, {w}}}

Out[92]= {637}
```

The variety of internal nesting in these representations can again be illustrated with …

```
In[93]:= Depth /@ list2Denested // (Short[#, 3]) &
```

```
Out[93]//Short= {2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4,
                 4, 4, 4, 3, 4, 4, 4, 4, 4, 4, 4, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4,
                 4, 4, 4, 4, 4, 3, 4, 4, 4, 3, 4, 4, 4, 3, 3, 4, 4, 4, 4, 4, 4, 3, 4,
                 <<505>>, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
                 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
                 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3}
```

Taking the **Union[]** of this result, and looking at the depths of all these members …

```
In[94]:= list2DenestedUnioned = Union [ list2Denested ];
         Depth /@ list2DenestedUnioned
         Dimensions @ %
```

```
Out[95]= {4, 5, 5, 5, 4, 5, 5, 5, 4, 4, 5, 5, 5, 5, 5, 5, 3, 4, 4, 4, 4, 4, 4, 4, 4, 5,
          5, 5, 4, 5, 5, 5, 4, 4, 5, 5, 5, 5, 5, 5, 3, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4,
          4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5,
          5, 5, 5, 5, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 3, 3, 4, 4, 4,
          4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 4, 3, 4, 4, 4, 3, 3, 4, 4, 4, 4,
          4, 4, 3, 4, 4, 4, 3, 4, 4, 4, 3, 3, 4, 4, 4, 4, 4, 4, 3, 3, 3, 3, 4, 4, 4,
          4, 4, 4, 4, 4, 4, 4, 4, 4, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3}

Out[96]= {176}
```

Since we now have the nested structures we will interpret as series-parallel circuits, we rename this list appropriately:

```
In[97]:= structureSet = list2DenestedUnioned;
         structureSet // Short
         Dimensions @ %
```

```
Out[98]//Short= {{t, {u, {v, w}}}, {t, {u, {v, {w}}}}, {t, {u, {{v}, w}}}, <<170>>,
                 {{t}, {u}, v, {w}}, {{t}, {u}, {v}, w}, {{t}, {u}, {v}, {w}}}

Out[99]= {176}
```

■ Section 4 - Nested Representations to Circuit Expressions.

Our nested structure representation is to be interpreted in two ways:

(i) a,b means series[a,b] and {a},{b} means paral[a,b], which we denote as the T1 transformations, and
(ii) a,b means paral[a,b] and {a},{b} means series[a,b], which we denote as the T2 transformations,

so we need to apply both T1 and T2 to our list of structures.

Here is how this in done. We choose one of the 176 members of **structureSet** at random (☺) …

```
In[100]:= structureSet[[100]]
```

```
Out[100]= {{t, u, {v}}, {w}}
```

… and see that it corresponds to the series-parallel expression …

```
In[101]:= nestStructureToResistanceT1[ structureSet[[100]] ]
```

```
Out[101]=
```

$$\frac{1}{\frac{1}{t+u+v}+\frac{1}{w}}$$

We will step through the internal code of **nestStructureToResistanceT1[**] to show the conversion process.

First we set the input variable to our example from **structureSet**;

```
In[102]:= originalTestExpression = structureSet[[100]]
```

```
Out[102]= {{t, u, {v}}, {w}}
```

The next step converts the nested expression to a list of character codes:

```
In[103]:= (* convert originalTestExpression to a list of character codes *)
          testExpression = ToCharacterCode[ ToString[ originalTestExpression ] ]
```

```
Out[103]= {123, 123, 116, 44, 32, 117, 44, 32,
           123, 118, 125, 125, 44, 32, 123, 119, 125, 125}
```

Character code 32 is a space, which *Mathematica* inserts automatically after commas; we want to remove these spaces, so we use ReplaceRepeated to drop all code 32's from the list:

```
In[104]:= (* remove any spaces *)
          testExpression = testExpression //. {x___, 32, y___} → {x, y}
```

```
Out[104]= {123, 123, 116, 44, 117, 44, 123, 118, 125, 125, 44, 123, 119, 125, 125}
```

```
In[105]:= FromCharacterCode[ testExpression ]
```

```
Out[105]= {{t,u,{v}},{w}}
```

The character codes for "{" and "}" are 123 and 125, and we want to replace these braces with left and right parentheses (codes 40 and 41):

```
In[106]:= (* convert all { to (, and all } to ) *)
          testExpression = testExpression //. {x___, 123, y___} → {x, 40, y};
           testExpression = testExpression //. {x___, 125, y___} → {x, 41, y};
          FromCharacterCode[ testExpression ]
```

```
Out[106]= ((t,u,(v)),(w)) Null²
```

As mentioned above, in our scheme an isolated "," is interpreted under the T1 transformation rules as "series", so checking with predefined Boolean tests to ensure that either or both of the enclosing symbols is not ")" or "(", the code for "," can be replaced by the codes for the letters of "~series~":

```
In[107]:= (* replace all "!),!(" with "~series~" *)
          testExpression =
            testExpression //. {u___, v_, w_?nonRbrQ, 44, x_?nonLbrQ, y_, z___} →
              {u, v, w, 126, 115, 101, 114, 105, 101, 115, 126, x, y, z} ;
          FromCharacterCode[ testExpression ]
```

```
Out[108]= ((t~series~u,(v)),(w))
```

```
In[109]:= (* replace all "),!(" with "~series~" *)
          testExpression =
            testExpression //. {u___, v_, w_?rbrQ, 44, x_?nonLbrQ, y_, z___} →
              {u, v, w, 126, 115, 101, 114, 105, 101, 115, 126, x, y, z} ;
          FromCharacterCode[ testExpression ]
```

```
Out[110]= ((t~series~u,(v)),(w))
```

```
In[111]:= (* replace all "!),(" with "~series~" *)
          testExpression =
            testExpression //. {u___, v_, w_?nonRbrQ, 44, x_?lbrQ, y_, z___} →
              {u, v, w, 126, 115, 101, 114, 105, 101, 115, 126, x, y, z};
          FromCharacterCode[testExpression]

Out[112]= ((t~series~u~series~(v)),(w))
```

Under the T1 transformation rules "},{" is interpreted as the parallel relationship, so such substrings can be replaced by the string "~paral~"

```
In[113]:= (* replace all "),(" with "~paral~" *)
          testExpression =
            testExpression //. {u___, v_, w_?rbrQ, 44, x_?lbrQ, y_, z___} →
              {u, v, w, 126, 112, 97, 114, 97, 108, 126, x, y, z};
          FromCharacterCode[testExpression]

Out[114]= ((t~series~u~series~(v))~paral~(w))
```

Running **`ToExpression[]`** on a string converts it to a *Mathematica* expression, i.e., executable *Mathematica* code, and the result of **`FromCharacterCode[testExpression]`** is a string corresponding to the list of character codes.

Since we have previously defined **`para[]`** and **`series[]`**, the resulting *Mathematica* expression is immediately evaluated, since it is recognized as using series and paral in Infix form, as illustrated in the following.

Note that the parentheses, which derived from the original braces, control the precedence of the various series and paral operators, and any "extra" parentheses (t), (u), etc., are ignored:

```
In[115]:= {series[xx, yy], series[series[xx, yy], zz],
           xx~series~yy~series~zz, (xx)~series~(yy)~series~zz}

Out[115]= {xx + yy, xx + yy + zz, xx + yy + zz, xx + yy + zz}

In[116]:= {paral[xx, yy], paral[paral[xx, yy], zz],
           xx~paral~yy~paral~zz, (xx)~paral~(yy)~paral~zz}

Out[116]= {1/(1/xx + 1/yy), 1/(1/xx + 1/yy + 1/zz), 1/(1/xx + 1/yy + 1/zz), 1/(1/xx + 1/yy + 1/zz)}
```

Recalling that we now have

```
In[117]:= FromCharacterCode[testExpression]

Out[117]= ((t~series~u~series~(v))~paral~(w))
```

The output from the final list of character codes, converted to a string and then a *Mathematica* expression, is …

```
In[118]:= (* convert testExpression from list of character codes to a string,
             and then to code expression which executes *)
          codeExpression = ToExpression[FromCharacterCode[testExpression]]

Out[118]= 1/(1/(t+u+v) + 1/w)
```

We can easily interpret this from the bottom up, as it were, as "t"-in-series-with-"u"-in-series-with"v", all in parallel with "w".

Applying **`nestStructureToResistanceT1[]`** to all the members of the list **`structureSet`**:

```
In[119]:= expressionListT1 =
            Union[ Map[ (nestStructureToResistanceT1[ # ]) &, structureSet ] ];
          expressionListT1 // Short
          Dimensions @ %
```

Out[120]//Short=

$$\left\{\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\ll 1\gg+\ll 1\gg}}},\ t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}},\ \frac{1}{\ll 1\gg+\ll 1\gg},\right.$$
$$\left.\ll 11\gg,\ \frac{1}{\ll 1\gg},\ \frac{1}{\frac{1}{t}+\frac{1}{u}+\frac{1}{\ll 1\gg}},\ \frac{1}{\frac{1}{t+u}+\frac{1}{v+w}},\ \frac{1}{\frac{1}{t}+\frac{1}{u+v+w}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v+w}}\right\}$$

```
Out[121]= {21}
```

Applying **nestStructureToResistanceT2[]** gives us another list of expressions:

```
In[122]:= expressionListT2 =
            Union[ Map[ (nestStructureToResistanceT2[ # ]) &, structureSet ] ];
          expressionListT2 // Short
          Dimensions @ %
```

Out[123]//Short=

$$\left\{\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\ll 1\gg+\ll 1\gg}}},\ \frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}},\right.$$
$$t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\ll 1\gg\ \ll 1\gg\ \ll 1\gg},\ \ll 11\gg,\ \ll 1\gg+w,$$
$$\left.\frac{1}{\frac{1}{t}+\frac{1}{\ll 1\gg}},\ \frac{1}{\frac{1}{t}+\frac{1}{u}+\frac{1}{v+w}},\ \frac{1}{\frac{1}{t}+\frac{1}{u+v+w}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v+w}}\right\}$$

```
Out[124]= {21}
```

We now combine these two list of candidate expressions, eliminating any possible literal duplicates by applying **Union[]** again …

```
In[125]:= expressionListT1T2 = Union[ expressionListT1 ~ Join ~ expressionListT2 ];
          expressionListT1T2 // Short
          Dimensions @ %
```

Out[126]//Short=

$$\left\{\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\ll 1\gg+\ll 1\gg}}},\ \frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\ll 1\gg},\right.$$
$$\ll 12\gg,\ \frac{1}{\ll 1\gg},\ \frac{1}{\frac{1}{t}+\ll 1\gg+\frac{1}{\ll 1\gg}},\ \frac{1}{\frac{1}{t+u}+\frac{1}{v+w}},\ \frac{1}{\frac{1}{t}+\frac{1}{u+v+w}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v+w}}\left.\right\}$$

```
Out[127]= {22}
```

■ Section 5 - Reduction to Unique Circuit Expressions.

We know, hope, or at least suspect, that this set of 22 expressions includes subsets of expressions that in fact represent the same circuit, i.e., they represent circuits that are topologically equivalent. Given the structure of the members of **expressionListT1T2** we need to construct a test for the topological equivalence of two circuit expressions. Some refinement of the original idea has resulted in some useful speed increases.

If two expressions do represents two circuits that are topologically i.e., electrically, equivalent, we detect this by finding that when a set of symbolic values for t, u, v, … is substituted into both expressions in all possible ways, there are common members in the two lists of resulting expressions.

```
In[128]:= perms = Permutations[ symbolSetAlternate ];
          nPerms = Length[ perms ] ;
          allRules = {};
          Do[ AppendTo[ allRules, Thread[ Rule[ symbolSet, perms[[i]] ] ] ],
            {i, 1, nPerms} ];
          allRules // Short
          Dimensions @ allRules

          Clear[ equivalentStructuresVersion2Q]
          equivalentStructuresVersion2Q[ struct1_, struct2_ ] :=
           Module[
            {allCircuitValues1, allCircuitValues2},
            allCircuitValues1 = Release[ Hold[struct1 ] /. allRules ];
            allCircuitValues2 = Release[ Hold[struct2 ] /. allRules ];
            Length[ Intersection[ allCircuitValues1, allCircuitValues2 ] ] > 0
           ]
```

```
Out[132]//Short= {{t → a, u → b, v → c, w → d}, {t → a, u → b, v → d, w → c},
                  {t → a, u → c, v → b, w → d}, <<18>>, {t → d, u → b, v → c, w → a},
                  {t → d, u → c, v → a, w → b}, {t → d, u → c, v → b, w → a}}

Out[133]= {24, 4}
```

We now use this custom **SameTest** to reduce **expressionListT1T2** to its topologically unique members.

```
In[136]:= Union[ expressionListT1T2,
             SameTest → equivalentStructuresVersion2Q] // Timing;
          temp = Last[%]
          Dimensions @ temp
          %%%[[1]]
```

$$\text{Out[137]= } \left\{\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}},\ \frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}},\right.$$
$$\left.\frac{1}{\frac{1}{t}+\frac{1}{u}+\frac{1}{v}+\frac{1}{w}},\ \frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}},\ \frac{1}{\frac{1}{t+u+v}+\frac{1}{w}},\ t+u+v+w,\ \frac{1}{\frac{1}{t+u}+\frac{1}{v+w}}\right\}$$

```
Out[138]= {10}

Out[139]= 0.591 Second
```

We find the correct number of possible 4-element circuits, as shown in the Table from Golinelli.

During algorithm development it was noticed that if the list **expressionListT1T2** is pre-processed as **Sort[(Sort[#])& /@ expressionListT1T2]**, all the essentially parallel circuits appear together, ahead of all the essentially series circuits. The price paid is that the **Union[]** operation is then slower.

```
In[140]:= Union[ Sort[ (Sort[#]) & /@ expressionListT1T2 ],
             SameTest → equivalentStructuresVersion2Q] // Timing;
          temp = Last[%]
          Dimensions @ temp
          %%%[[1]]
```

$$\text{Out[141]= } \left\{(-1)^{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}},\ (-1)^{\frac{1}{t}+\frac{1}{u}+\frac{1}{v}+\frac{1}{w}},\ (-1)^{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}},\ (-1)^{\frac{1}{t+u+v}+\frac{1}{w}},\ (-1)^{\frac{1}{t+u}+\frac{1}{v+w}},\right.$$
$$\left.\frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}},\ t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}},\ t+u+v+w\right\}$$

```
Out[142]= {10}

Out[143]= 1.202 Second
```

The essentially parallel circuit expressions have been modified. The application of **repairFunction[]** "repairs" the effect of **Sort[]** applied to individual essentially parallel circuit expressions.

```
In[144]:= uniqueStructures10 = (repairFunction[#]) & /@ temp
```

```
Out[144]= {1/(1/t + 1/(u + 1/(1/v + 1/w))), 1/(1/t + 1/u + 1/v + 1/w), 1/(1/(t+u) + 1/v + 1/w), 1/(1/(t+u+v) + 1/w), 1/(1/(t+u) + 1/(v+w)),
  1/(1/t + 1/u) + 1/(1/v + 1/w), t + u + 1/(1/v + 1/w), t + 1/(1/u + 1/v + 1/w), t + 1/(1/(u+v) + 1/w), t + u + v + w}
```

In terms of the *Mathematica* FullForms for these circuit expressions they are now in a canonical order, and it is interesting to note that the 5 essentially parallel circuits are listed first, and then the 5 essentially series circuits. Referring to the 4-element circuits in Golinelli's table as numbers 1 to 5 down each column, **uniqueStructures10** has the essentially series circuits the order {5, 2, 3, 4, 1}, and the essentially parallel circuits in the order {4, 1, 2, 3, 5}.

■ Section 6 - How Many Circuits Actually Are There?

The following code (slightly reformatted) is from page 209 of the Symbolics volume of Michael Trott's magnificent set of *Mathematica* GuideBooks [7 ★].

```
In[145]:= seriesParallelNetworks[ n_ ] :=
 Module[ {S, τ},
  Rest[ (#[[3]]) & @ Fold[
      ( #1 / (1 - S^#2)^SeriesCoefficient[##] ) &,
      1 / (1 - (S = τ + O[τ]^(n+1))), Range[2, n] ] ] ]
```

This list the number of i-element circuits for i = 1, … , n:

```
In[146]:= seriesParallelNetworks[4]
```

```
Out[146]= {1, 2, 4, 10}
```

This is series A000084 in [6 ★].

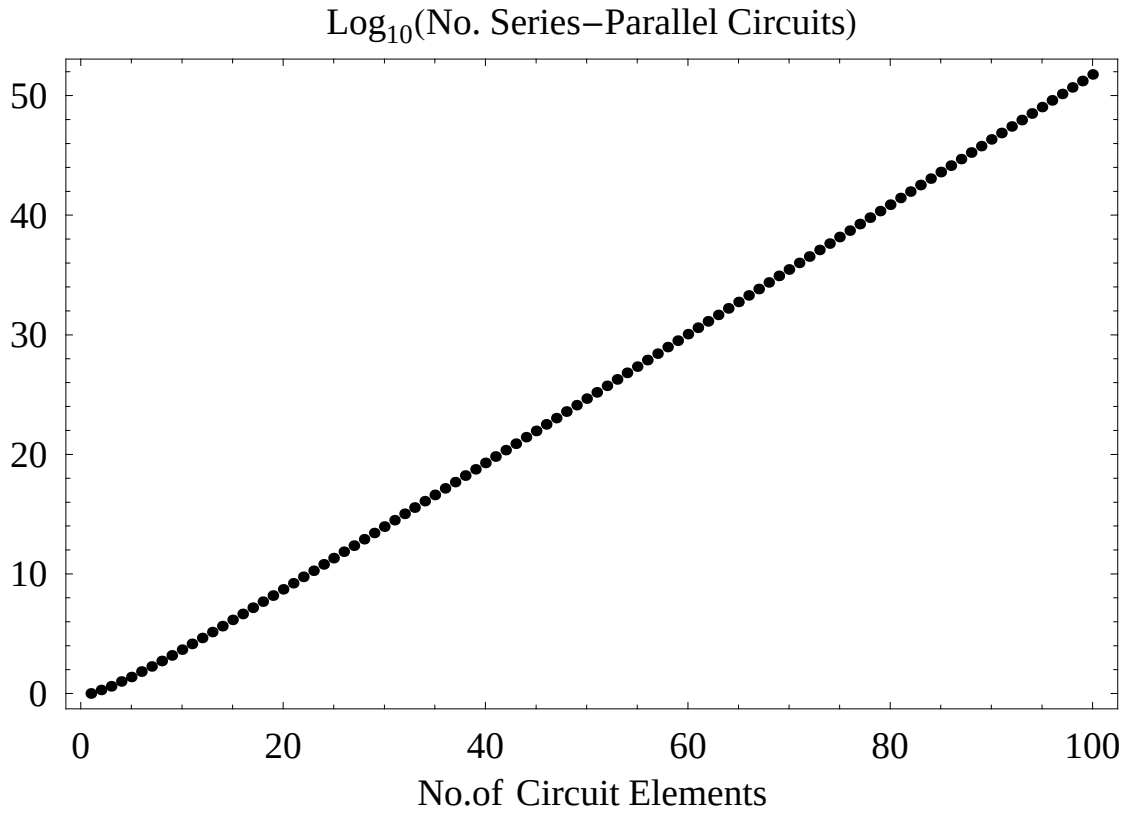
How fast do these numbers grow?

In[147]:=
```
seriesParallelNetworks[100]
```

Out[147]=
```
{1, 2, 4, 10, 24, 66, 180, 522, 1532, 4624, 14136, 43930, 137908,
 437502, 1399068, 4507352, 14611576, 47633486, 156047204, 513477502,
 1696305728, 5623993944, 18706733128, 62408176762, 208769240140,
 700129713630, 2353386723912, 7927504004640, 26757247573360,
 90479177302242, 306481637180676, 1039825579149516, 3533238105431144,
 12022695376246364, 40964814174562504, 139754932965689486,
 477353256699920544, 1632304839666178522, 5587602984719422092,
 19146480336548243722, 65670379223051730824, 225448266880936115440,
 774644024342635183112, 2663899039964678479766, 9168056380423173365752,
 31576604401276670869996, 108834834444071971878620,
 375380216090153021947396, 1295577092445615344707920,
 447436890568679579943934, 1546200807955687453594496,
 534632268293697284571027 04, 184965098681578137529520416,
 640266088356689438891182328, 2217474110093931339894990256,
 768379368433160383369123 5214, 2663815169835907114813202 9156,
 923924159830316391516599873 52, 3206013703442119428153927674 60,
 111297284592725488385063576 4284, 386533882696858118786805012 7084,
 134297630543788515711653084 42446, 466789606991879803552118750 20116,
 162308248470152158059930751323490, 564574664201162207415152351970584,
 196452968019533106989959613 8296600, 683828947331705011371588991 1528612,
 238113225319320482805154378582748 16,
 829396841981550102692244686893850 32,
 288988074405687450992788693020570268,
 100723819850186726666555380475007 7044,
 351168362444148143567345806851934 8734,
 122468914817102027164165640642161 15128,
 427228447110738713610909665304648 97756,
 149078466258987168672505416658403914924,
 520339647544524031823456285642256277566,
 181665762128746206303283126927339 4808864,
 634410630466205476598764439563070 0540790,
 221603271363655023673793173323150 99839744,
 774261188501269935083587404993378 10322424,
 270583792742485609857220323267798421284832,
 945837428376917070860445149486993106516008,
 330696187931611654738260028286951 0026487680,
 115647832713163203650207403036416915539978 76,
 404519208546354188320592374843956818316222 32,
 141524602779621661487309968280622652779830584,
 495237778204087438163764102409182103155783836,
 173333524808382664387079292791844030518268 8268,
 606787149463205177847369245174205689153115 8768,
 212458091093281564840558003398388830097247 96736,
 744031673234288037026841228758972760523596 12520,
 260608775437880246546580544642700506123589989184,
 912986558309579051786333684057977832331820841240,
 319901146474748451139206104322552166273991071 4790,
 112109304038180307824803656433932563056785797 11152,
 392952808159354630871610827395343636915307696 02580,
 137755992221127539795621877673326337524346188 531108,
 483003739444661173263343449222119429018103997 875932,
 169378761281456349065713099083714179960687051 8096788,
 594065722598839086522569396158486094463250101066 1006}
```

It's not immediately clear just how big these numbers are. Plotting them on a Log_{10}scale …

```
In[148]:= ListPlot[ Log[ 10, seriesParallelNetworks[100] ],
            PlotStyle → AbsolutePointSize[4],
            PlotRange → All, Axes → None, Frame → True,
            FrameLabel → {"No.of Circuit Elements", None}, RotateLabel → False,
            PlotLabel -> "Log10(No. Series-Parallel Circuits)",
            TextStyle → {FontFamily → "Times", FontSize → 14}, ImageSize → 500 ];
```



…we recall that the number of atoms making up planet Earth is usually estimated to be around 10^{50}. The growth rate is of the order of …

```
In[149]:= circuits /. Solve[ Log[10, circuits] / n == 52. / 100, circuits ][[1]]

Out[149]= e^(1.19734 n)
```

■ Section 7 - Finding the Possible Total Resistances - 4 Elements.

Returning to the above results stored in the list **uniqueStructures10**, it is fairly clear that if we were to try substituting a given set of 4 resistors into all of the positions in a given single circuit *in all possible ways*, which we can do in 4!=24 ways, then potentially a number of distinct total resistance values may be obtained. Of course interchanging two or more resistors that are all in parallel or series will not change their total resistance, but in general several distinct total resistances will be produced from each circuit.

We choose a set of values, create a set of rules which substitute the values in all possible ways into the 10 structures, and finally produce a sorted list of total values, the circuit which gives each value, the positions into which the chosen values must be placed, and an index number for reference.

We now want to see how many different total resistances we can obtain with these 10 4-resistor circuits. We choose some prime individual resistor values …

```
In[150]:= selectedValues = {2, 3, 5, 7};
```

… and generate some value substitution rules …

```
In[151]:= perms = Permutations[ selectedValues ];
          nPerms = Length[ perms ] ;
          allRules = {};
          Do[ AppendTo[ allRules, Thread[ Rule[ symbolSet, perms[[i]] ] ] ],
            {i, 1, nPerms} ];
          allRules // Short
          Dimensions @ allRules
```

```
Out[155]//Short= {{t → 2, u → 3, v → 5, w → 7}, {t → 2, u → 3, v → 7, w → 5},
                  {t → 2, u → 5, v → 3, w → 7}, <<18>>, {t → 7, u → 3, v → 5, w → 2},
                  {t → 7, u → 5, v → 2, w → 3}, {t → 7, u → 5, v → 3, w → 2}}

Out[156]= {24, 4}
```

Substituting the chosen values into all 10 circuits in all possible ways …

```
In[157]:= uniqueStructuresValues10 = uniqueStructures10 /. allRules // N;
          uniqueStructuresValues10 // Short
          Dimensions @ uniqueStructuresValues10
```

```
Out[158]//Short= {{1.49474, 0.850202, 1.84211, <<4>>, 3.47887, 5.73333, 17.},
                  <<22>>, {<<1>>}}

Out[159]= {24, 10}
```

```
In[160]:= uniqueStructuresValues10 = Transpose[ uniqueStructuresValues10 ];
```

```
In[161]:= valuesAndRules =
            (Transpose[ { #, allRules} ]) & /@ uniqueStructuresValues10;
```

```
In[162]:= repeatedCircuits =
            Flatten[ Transpose[ Table[ uniqueStructures10, {nPerms} ] ] ];
          Dimensions @
           %
```

```
Out[163]= {240}
```

This function interchanges the first two elements in a list.

```
In[164]:= Clear[ swapList1and2 ]
          swapList1and2[ list_ ] :=
           Reverse[ Take[ list, 2 ] ] ~Join~ Take[ list, -(Length[ list ] - 2) ]
```

```
In[166]:= usefulData = Sort[ (swapList1and2[ # ]) & /@ ((Flatten[#, 1]) & /@
                Transpose[ {repeatedCircuits, Flatten[ valuesAndRules, 1 ]} ]) ];
          usefulData // Short
          Dimensions @ usefulData
```

```
Out[167]//Short= {{0.850202, 1/(1/t + 1/u + 1/v + 1/w), {t → 2, u → 3, v → 5, w → 7}},
                  <<238>>, {17., t + u + v + w, {t → 7, u → 5, v → 3, w → 2}}}

Out[168]= {240, 3}
```

This Boolean test compares lists on the basis of the first elements:

```
In[169]:= Clear[ sameCircuitValueQ ]
          sameCircuitValueQ[ x_, y_ ] := First[ x ] === First[ y ]
```

Remove circuit arrangements with duplicated total resistances:

```
In[171]:= reallyUsefulData = Union[ usefulData , SameTest → sameCircuitValueQ ];
          reallyUsefulData // Short
          Dimensions @ reallyUsefulData
```

```
Out[172]//Short= {{0.850202, 1/(1/t + 1/u + 1/v + 1/w), {t → 2, u → 3, v → 5, w → 7}},
                  <<46>>, {17., t + u + v + w, {t → 2, u → 3, v → 5, w → 7}}}

Out[173]= {48, 3}
```

In this last step we add an index number to each entry:

```
In[174]:= reallyUsefulDataIndexed = Map[ (Flatten[#, 1]) &,
             Transpose[ {reallyUsefulData, Range[ Length[ reallyUsefulData ] ]} ] ];
          reallyUsefulDataIndexed // Short
```

```
Out[175]//Short= {{0.850202, 1/(1/t + 1/u + 1/v + 1/w), {t → 2, u → 3, v → 5, w → 7}, 1},
                  <<46>>, {17., t + u + v + w, {t → 2, u → 3, v → 5, w → 7}, 48}}
```

Displaying the 48 results in a form useful for look-up:

```
In[176]:= reallyUsefulDataIndexed // MatrixForm
```

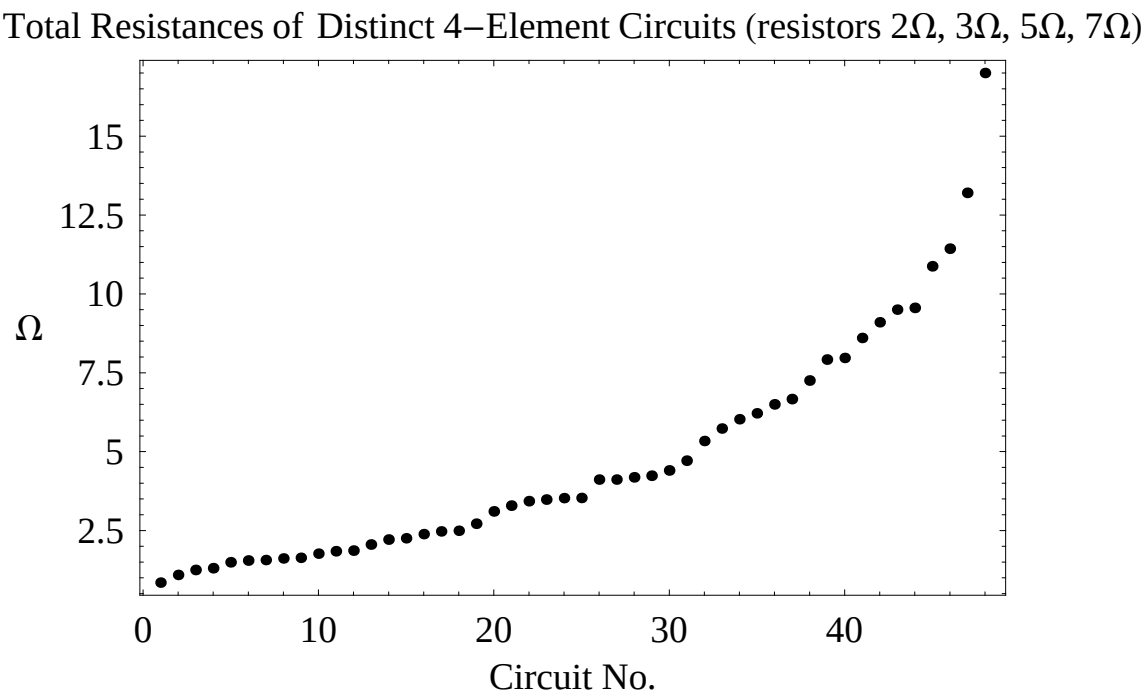
0.850202	$\frac{1}{\frac{1}{t}+\frac{1}{u}+\frac{1}{v}+\frac{1}{w}}$	{t → 2, u → 3, v → 5, w → 7}	1
1.09091	$\frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}}$	{t → 5, u → 7, v → 2, w → 3}	2
1.25	$\frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}}$	{t → 3, u → 7, v → 2, w → 5}	3
1.30233	$\frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}}$	{t → 3, u → 5, v → 2, w → 7}	4
1.49474	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 2, u → 3, v → 5, w → 7}	5
1.55172	$\frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}}$	{t → 2, u → 7, v → 3, w → 5}	6
1.56044	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 2, u → 5, v → 3, w → 7}	7
1.61538	$\frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}}$	{t → 2, u → 5, v → 3, w → 7}	8
1.63218	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 2, u → 7, v → 3, w → 5}	9
1.76471	$\frac{1}{\frac{1}{t+u+v}+\frac{1}{w}}$	{t → 3, u → 5, v → 7, w → 2}	10
1.84211	$\frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}}$	{t → 2, u → 3, v → 5, w → 7}	11
1.86316	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 3, u → 2, v → 5, w → 7}	12
2.05814	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 3, u → 5, v → 2, w → 7}	13
2.2125	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 3, u → 7, v → 2, w → 5}	14
2.25275	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 5, u → 2, v → 3, w → 7}	15
2.38372	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 5, u → 3, v → 2, w → 7}	16
2.47059	$\frac{1}{\frac{1}{t+u+v}+\frac{1}{w}}$	{t → 2, u → 5, v → 7, w → 3}	17
2.49425	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 7, u → 2, v → 3, w → 5}	18
2.7125	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 7, u → 3, v → 2, w → 5}	19
3.10606	$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}$	{t → 5, u → 7, v → 2, w → 3}	20
3.28788	$\frac{1}{\frac{1}{t}+\frac{1}{}}$	{t → 7, u → 5, v → 2, w → 3}	21

Out[176]//MatrixForm=

3.43056	$\frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 2, u\to 7, v\to 3, w\to 5\}$	22
3.47887	$t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}}$	$\{t\to 2, u\to 3, v\to 5, w\to 7\}$	23
3.52857	$\frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 2, u\to 5, v\to 3, w\to 7\}$	24
3.52941	$\frac{1}{\frac{1}{t+u+v}+\frac{1}{w}}$	$\{t\to 2, u\to 3, v\to 7, w\to 5\}$	25
4.11667	$\frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 2, u\to 3, v\to 5, w\to 7\}$	26
4.11765	$\frac{1}{\frac{1}{t+u+v}+\frac{1}{w}}$	$\{t\to 2, u\to 3, v\to 5, w\to 7\}$	27
4.18644	$t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}}$	$\{t\to 3, u\to 2, v\to 5, w\to 7\}$	28
4.23529	$\frac{1}{\frac{1}{t+u}+\frac{1}{v+w}}$	$\{t\to 2, u\to 7, v\to 3, w\to 5\}$	29
4.4	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 2, u\to 5, v\to 7, w\to 3\}$	30
4.71429	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 3, u\to 5, v\to 7, w\to 2\}$	31
5.33333	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 2, u\to 3, v\to 7, w\to 5\}$	32
5.73333	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 2, u\to 3, v\to 5, w\to 7\}$	33
6.02439	$t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}}$	$\{t\to 5, u\to 2, v\to 3, w\to 7\}$	34
6.21429	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 3, u\to 2, v\to 7, w\to 5\}$	35
6.5	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 3, u\to 2, v\to 5, w\to 7\}$	36
6.66667	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 5, u\to 3, v\to 7, w\to 2\}$	37
7.25	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 5, u\to 2, v\to 7, w\to 3\}$	38
7.91667	$t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 2, u\to 3, v\to 5, w\to 7\}$	39
7.96774	$t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}}$	$\{t\to 7, u\to 2, v\to 3, w\to 5\}$	40
8.6	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 7, u\to 3, v\to 5, w\to 2\}$	41
9.1	$t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 2, u\to 5, v\to 3, w\to 7\}$	42
9.5	$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}$	$\{t\to 7, u\to 2, v\to 3, w\to 5\}$	43
9.55556	$t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 3, u\to 5, v\to 2, w\to 7\}$	44
10.875	$t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 2, u\to 7, v\to 3, w\to 5\}$	45
11.4286	$t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 3, u\to 7, v\to 2, w\to 5\}$	46
13.2	$t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}}$	$\{t\to 5, u\to 7, v\to 2, w\to 3\}$	47
17.	$t+u+v+w$	$\{t\to 2, u\to 3, v\to 5, w\to 7\}$	48

A simple ListPlot indicates how these total resistance values are distributed between the minimum (all in parallel) and the maximum (all series):

In[177]:=
```
alldistinctValues = Map[ (First[#]) &, reallyUsefulDataIndexed ];
ListPlot[ alldistinctValues,
  PlotStyle → AbsolutePointSize[4], PlotRange → All,
  Axes → None, Frame → True, FrameLabel → {"Circuit No.", "Ω"},
  RotateLabel → False, PlotLabel -> "Total Resistances of
     Distinct 4-Element Circuits (resistors 2Ω, 3Ω, 5Ω, 7Ω)",
  TextStyle → {FontFamily → "Times", FontSize → 14}, ImageSize → 500 ];
```



■ Section 8 - 4-Element Circuit Density.

Suppose we were interested in fabricating a net resistance of around 4 Ω. From the above list we can select some candidate circuits in a range around 4 Ω:

In[179]:=
```
desiredResistanceRange = {3.5, 4.5};
Select[ reallyUsefulDataIndexed, (desiredResistanceRange[[1]] ≤ #[[1]] &&
     #[[1]] ≤ desiredResistanceRange[[2]]) & ];
% // MatrixForm
```

Out[181]//MatrixForm=

$$\begin{pmatrix} 3.52857 & \frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}} & \{t\to 2,\ u\to 5,\ v\to 3,\ w\to 7\} & 24 \\ 3.52941 & \frac{1}{\frac{1}{t+u+v}+\frac{1}{w}} & \{t\to 2,\ u\to 3,\ v\to 7,\ w\to 5\} & 25 \\ 4.11667 & \frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}} & \{t\to 2,\ u\to 3,\ v\to 5,\ w\to 7\} & 26 \\ 4.11765 & \frac{1}{\frac{1}{t+u+v}+\frac{1}{w}} & \{t\to 2,\ u\to 3,\ v\to 5,\ w\to 7\} & 27 \\ 4.18644 & t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}} & \{t\to 3,\ u\to 2,\ v\to 5,\ w\to 7\} & 28 \\ 4.23529 & \frac{1}{\frac{1}{t+u}+\frac{1}{v+w}} & \{t\to 2,\ u\to 7,\ v\to 3,\ w\to 5\} & 29 \\ 4.4 & t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}} & \{t\to 2,\ u\to 5,\ v\to 7,\ w\to 3\} & 30 \end{pmatrix}$$

These 6 circuits can be seen to have 5 different topologies (25 and 27 are equivalent, but have the values for "v" and "w" interchanged), and the question arises as to which circuit is most robust to variations in individual resistor values. The code below was run for each of the 7 circuits in the

required resistance range. All the results are given after the code.To illustrate the process, we asses the robustness of …

```
In[182]:= circuitNumber = 24;
          testStructure = reallyUsefulDataIndexed[[circuitNumber, 2]]
```

Out[183]= $\frac{1}{\frac{1}{t}+\frac{1}{u}}+\frac{1}{\frac{1}{v}+\frac{1}{w}}$

… with the substituted values …

```
In[184]:= reallyUsefulDataIndexed[[circuitNumber, 3]]
```

```
Out[184]= {t → 2, u → 5, v → 3, w → 7}
```

```
In[185]:= substitutedValues =
           symbolSet /. reallyUsefulDataIndexed[[circuitNumber, 3]]
```

```
Out[185]= {2, 5, 3, 7}
```

… and resistors with tolerance expressed as a % Normal distribution standard deviation of …

```
In[186]:= resistorQuality = 0.05;
```

We generate a list of **nSamples** total circuit resistances in the spirit of a Monte Carlo approach to assess the robustness of this circuit …

```
In[187]:= nSamples = 2000;
          randomResistorValues = Table[
             Map[ (Random[#]) &, Thread[ NormalDistribution[ substitutedValues,
                resistorQuality * substitutedValues ] ] ], {nSamples} ];
          randomResistorValues // Short
```

```
Out[189]//Short= {{1.95274, 5.00769, 3.02549, 7.13956}, <<1998>>, <<1>>}
```

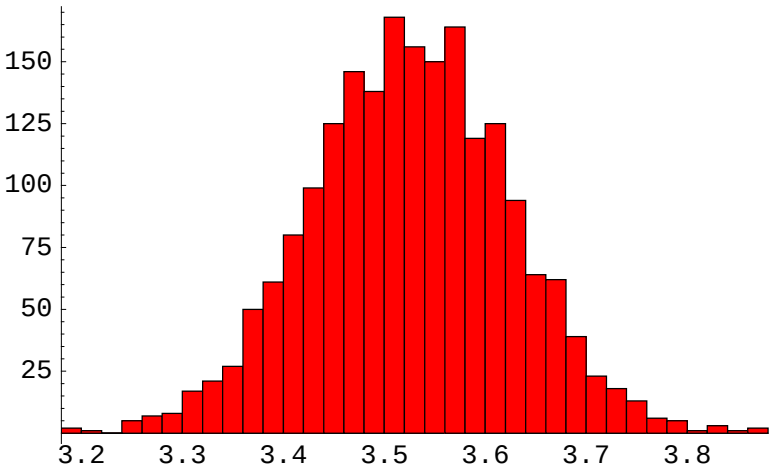
```
In[190]:= distributedValueRules = {};
          Do[ AppendTo[ distributedValueRules,
            Thread[ Rule[ symbolSet, randomResistorValues[[i]] ] ] ], {i, 1, nSamples} ]
          distributedValueRules // Short
```

```
Out[192]//Short= {{t → 1.95274, u → 5.00769, v → 3.02549, w → 7.13956}, <<1999>>}
```

```
In[193]:= distributedTotalResistances = testStructure /. distributedValueRules;
```

Not surprisingly, the distribution of the total circuit resistance appears to be close to Normal:

```
In[194]:= Histogram[ distributedTotalResistances ];
```



```
In[195]:= Print["distribution mean = ", distributionMean =
            Mean /. LocationReport[distributedTotalResistances], " \n",
           "average % error = ",
           (distributionMean - reallyUsefulDataIndexed[[circuitNumber, 1]]) /
             reallyUsefulDataIndexed[[circuitNumber, 1]] * 100 "%", " \n",
           "standard deviation = ", sd = StandardDeviation /.
            DispersionReport[distributedTotalResistances], " \n",
           "standard deviation of an\nequivalent quality single resistor = ",
           singleResistorSD =
            reallyUsefulDataIndexed[[circuitNumber, 1]] * resistorQuality, " \n",
           "sd reduction factor = ", sd / singleResistorSD
          ];
```

```
distribution mean = 3.52572
average % error = -0.0808637 %
standard deviation = 0.0983113
standard deviation of an
equivalent quality single resistor = 0.176429
sd reduction factor = 0.55723
```

********** Summary results for circuit no. 24:

```
distribution mean = 3.53042
average % error = 0.0285079 %
standard deviation = 0.129001
standard deviation of an
equivalent quality single resistor = 0.176471
sd reduction factor = 0.731003
```

********** Summary results for circuit no. 25:

```
distribution mean = 4.10805
average % error = -0.209192 %
standard deviation = 0.112287
standard deviation of an
equivalent quality single resistor = 0.205833
sd reduction factor = 0.545522
```

********** Summary results for circuit no. 26:

```
distribution mean = 4.11633
average % error = -0.0319596 %
standard deviation = 0.134917
standard deviation of an
equivalent quality single resistor = 0.205882
sd reduction factor = 0.655314
```

********** Summary results for circuit no. 27:

```
distribution mean = 4.18103
average % error = -0.129215 %
standard deviation = 0.159149
standard deviation of an
equivalent quality single resistor = 0.209322
sd reduction factor = 0.760305
```

********** Summary results for circuit no. 28:

```
distribution mean = 4.22706
average % error = -0.194498 %
standard deviation = 0.112905
standard deviation of an
equivalent quality single resistor = 0.211765
sd reduction factor = 0.533161
```

********** Summary results for circuit no. 29:

```
distribution mean = 4.39247
average % error = -0.171194 %
standard deviation = 0.138251
standard deviation of an
equivalent quality single resistor = 0.22
sd reduction factor = 0.628415
```

********** Summary results for circuit no. 30:

From this small amount of data, there appears to be a phenomenon in which (i) the total circuit resistance is biased downwards, and (ii) there is an "sd reduction factor" of between 0.5 and 0.8. This suggests that variation of the total resistance of circuits comprising several resistors in a series-parallel configuration is somewhat reduced compared with variation in individual resistors.

If we look at some simple derivatives …

In[196]:=
$$\partial_t \left(\frac{1}{\frac{1}{t}+\frac{1}{u}} + \frac{1}{\frac{1}{v}+\frac{1}{w}} \text{ /. } \{u \to 3,\ v \to 5,\ w \to 7\} \right) \text{ /. } \{t \to 2\}$$

Out[196]=
$$\frac{9}{25}$$

In[197]:=
$$\partial_u \left(\frac{1}{\frac{1}{t}+\frac{1}{u}} + \frac{1}{\frac{1}{v}+\frac{1}{w}} \text{ /. } \{t \to 2,\ v \to 5,\ w \to 7\} \right) \text{ /. } \{u \to 3\}$$

Out[197]=
$$\frac{4}{25}$$

In[198]:=
$$\partial_v \left(\frac{1}{\frac{1}{t}+\frac{1}{u}} + \frac{1}{\frac{1}{v}+\frac{1}{w}} \text{ /. } \{t \to 2,\ u \to 3,\ w \to 7\} \right) \text{ /. } \{v \to 5\}$$

Out[198]=
$$\frac{49}{144}$$

In[199]:=
$$\partial_w \left(\frac{1}{\frac{1}{t}+\frac{1}{u}} + \frac{1}{\frac{1}{v}+\frac{1}{w}} \text{ /. } \{t \to 2,\ u \to 3,\ v \to 5\} \right) \text{ /. } \{w \to 7\}$$

Out[199]=
$$\frac{25}{144}$$

… there is some support for this idea, as all these first derivatives are less than unity.

■ Section 9 - What about 5 Resistors?

In[200]:=
```
seriesParallelNetworks[5]
```

Out[200]=
```
{1, 2, 4, 10, 24}
```

There are 24 circuits. In Figure 11, taken from Lomnicki's [4 ★], Table 1 p.113, the possible 5-element circuits are listed, with the essentially series circuits in the left column and the essentially parallel on the right.

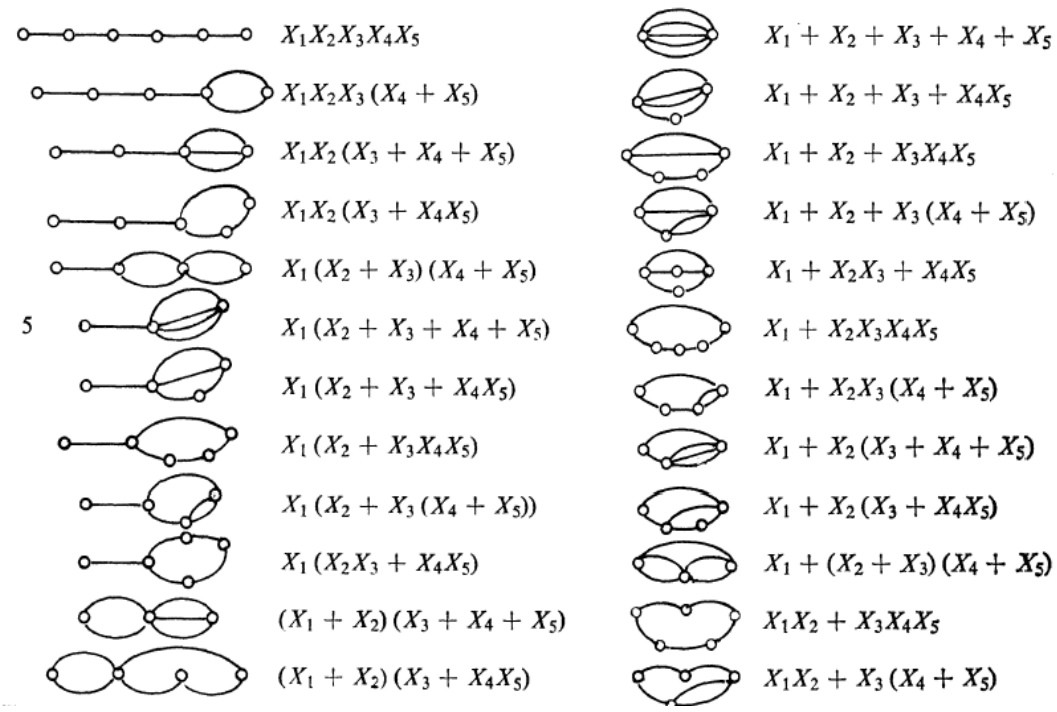


Figure 11. Table 1 from Lomnicki [4 ✱]

Our results:

```
uniqueStructures24 = (repairFunction[#]) & /@ temp
Dimensions @ uniqueStructures24
```

$$\left\{\frac{1}{\frac{1}{t}+\frac{1}{\frac{1}{\frac{1}{u}+\frac{1}{v}}+\frac{1}{\frac{1}{w}+\frac{1}{x}}}}, \frac{1}{\frac{1}{t}+\frac{1}{u}+\frac{1}{v+\frac{1}{\frac{1}{w}+\frac{1}{x}}}}, \frac{1}{\frac{1}{t+u}+\frac{1}{v+\frac{1}{\frac{1}{w}+\frac{1}{x}}}}, \frac{1}{\frac{1}{t}+\frac{1}{u+v+\frac{1}{\frac{1}{w}+\frac{1}{x}}}}\right.,$$

$$\frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}+\frac{1}{x}}}}, \frac{1}{\frac{1}{t}+\frac{1}{u+\frac{1}{\frac{1}{v+w}+\frac{1}{x}}}}, \frac{1}{\frac{1}{t}+\frac{1}{u}+\frac{1}{v}+\frac{1}{w}+\frac{1}{x}}, \frac{1}{\frac{1}{t+u}+\frac{1}{v}+\frac{1}{w}+\frac{1}{x}},$$

$$\frac{1}{\frac{1}{t+u+v}+\frac{1}{w}+\frac{1}{x}}, \frac{1}{\frac{1}{t+u}+\frac{1}{v+w}+\frac{1}{x}}, \frac{1}{\frac{1}{t+u+v+w}+\frac{1}{x}}, \frac{1}{\frac{1}{t+u+v}+\frac{1}{w+x}}, t+\frac{1}{\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}}+\frac{1}{x}},$$

$$t+\frac{1}{\frac{1}{u}+\frac{1}{v}}+\frac{1}{\frac{1}{w}+\frac{1}{x}}, \frac{1}{\frac{1}{t}+\frac{1}{u}+\frac{1}{v}}+\frac{1}{\frac{1}{w}+\frac{1}{x}}, \frac{1}{\frac{1}{t+u}+\frac{1}{v}}+\frac{1}{\frac{1}{w}+\frac{1}{x}},$$

$$t+u+v+\frac{1}{\frac{1}{w}+\frac{1}{x}}, t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}+\frac{1}{x}}, t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}+\frac{1}{x}}, t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}+\frac{1}{x}},$$

$$\left.t+u+\frac{1}{\frac{1}{v+w}+\frac{1}{x}}, t+\frac{1}{\frac{1}{u+v+w}+\frac{1}{x}}, t+u+v+w+x, \frac{1}{\frac{1}{t+u}+\frac{1}{v+w}}+x\right\}$$

```
{24}
```

Our circuit expressions are again in a canonical order, with the 12 essentially parallel circuits listed first. Referring to the 5-element circuits in Lomnicki's table as numbers 1 to 12 down each column, **uniqueStructures24** has the essentially series circuits in the order {9, 5, 11, 12, 2, 3, 6, 7, 4, 8, 1, 10}, and the essentially parallel circuits the order {10, 4, 12, 7, 8, 9, 1, 2, 3, 5, 6, 11}.

■ Section 15 - What about 6 Resistors? A Surprise …

We omit the code for the generation of all possible 6-element circuits, which apart from the following set-up of **symbolSet** and **symbolSetAlternate** is identical to that used above.

```
symbolSet = {t, u, v, w, x, y};
      Clear[t, u, v, w, x, y]
symbolSetAlternate = {a, b, c, d, e, f};
               Clear[a, b, c, d, e, f]
```

The results are somewhat surprising. Here explicitly are all the 6-element circuits found by the our approach, with the essentially parallel circuits listed ahead of the essentially series circuits.

In[201]:= **circuitsWith6Elements** =

$$\left\{\frac{1}{\frac{1}{u}+\frac{1}{v+\frac{1}{\frac{1}{w+\frac{1}{x}}+\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}}, \frac{1}{\frac{1}{u}+\frac{1}{v+\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}}, \frac{1}{\frac{1}{u}+\frac{1}{\frac{1}{v}+\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}}},\right.$$

$$\frac{1}{\frac{1}{u}+\frac{1}{\frac{1}{v+w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}}, \frac{1}{\frac{1}{u+\frac{1}{\frac{1}{v}+\frac{1}{w}}}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}},$$

$$\frac{1}{\frac{1}{u+v+w}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w+x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w+x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v+w+x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}},$$

$$\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w+\frac{1}{\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w+\frac{1}{\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v+w+\frac{1}{\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v+\frac{1}{\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}}},$$

$$\frac{1}{\frac{1}{u}+\frac{1}{v+\frac{1}{\frac{1}{w+x}+\frac{1}{y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w+\frac{1}{\frac{1}{x+y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w+\frac{1}{\frac{1}{x+y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v+w+\frac{1}{\frac{1}{x+y}+\frac{1}{z}}}},$$

$$\frac{1}{\frac{1}{u}+\frac{1}{v+\frac{1}{\frac{1}{w+x+y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},$$

$$\frac{1}{\frac{1}{u+v+w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w+x}+\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u+v+w+x}+\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u+v+w}+\frac{1}{x+y}+\frac{1}{z}},$$

$$\frac{1}{\frac{1}{u+v+w+x+y}+\frac{1}{z}}, \frac{1}{\frac{1}{\frac{1}{u+v}+\frac{1}{w+x}}+y+\frac{1}{z}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w+x}+\frac{1}{y+z}}, \frac{1}{\frac{1}{u+v+w+x}+\frac{1}{y+z}},$$

$$\frac{1}{\frac{1}{u+v+w}+\frac{1}{x+y+z}}, u+\frac{1}{\frac{1}{v}+\frac{1}{\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}}, \frac{1}{\frac{1}{u}+\frac{1}{v}}+\frac{1}{\frac{1}{w}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, u+v+\frac{1}{\frac{1}{w}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}},$$

$$u+\frac{1}{\frac{1}{v}+\frac{1}{w}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, u+\frac{1}{\frac{1}{v+w}+\frac{1}{x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, u+\frac{1}{\frac{1}{v}+\frac{1}{w+x+\frac{1}{\frac{1}{y}+\frac{1}{z}}}}, u+\frac{1}{\frac{1}{v}+\frac{1}{w+\frac{1}{\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}}},$$

$$u+\frac{1}{\frac{1}{v}+\frac{1}{w+\frac{1}{\frac{1}{x+y}+\frac{1}{z}}}}, \frac{1}{\frac{1}{u}+\frac{1}{v}}+\frac{1}{\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}, u+v+\frac{1}{\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}},$$

$$u+\frac{1}{\frac{1}{v}+\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{v+w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}},$$

$$u+\frac{1}{\frac{1}{v+w}+\frac{1}{x}}+\frac{1}{\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}}+\frac{1}{\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w}}+\frac{1}{\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}, u+v+w+\frac{1}{\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},$$

$$u+v+\frac{1}{\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}, u+\frac{1}{\frac{1}{v}+\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}}, u+\frac{1}{\frac{1}{v+w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},$$

$$u+v+\frac{1}{\frac{1}{w+x}+\frac{1}{y}+\frac{1}{z}}, u+\frac{1}{\frac{1}{v+w+x}+\frac{1}{y}+\frac{1}{z}}, \frac{1}{\frac{1}{u+v}+\frac{1}{w}}+\frac{1}{\frac{1}{x+y}+\frac{1}{z}},$$

$$u+v+w+\frac{1}{\frac{1}{x+y}+\frac{1}{z}}, u+\frac{1}{\frac{1}{v+w}+\frac{1}{x+y}+\frac{1}{z}}, u+v+\frac{1}{\frac{1}{w+x+y}+\frac{1}{z}},$$

$$\left.u+\frac{1}{\frac{1}{v+w+x+y}+\frac{1}{z}}, u+v+w+x+y+z, \frac{1}{\frac{1}{u+v}+\frac{1}{w+x}}+y+z, \frac{1}{\frac{1}{u+v+w}+\frac{1}{x+y}}+z\right\};$$

In[202]:=
```
Dimensions @ circuitsWith6Elements
```

Out[202]=
```
{64}
```

We found 64 circuits, but …

In[203]:=
```
seriesParallelNetworks[6]
```

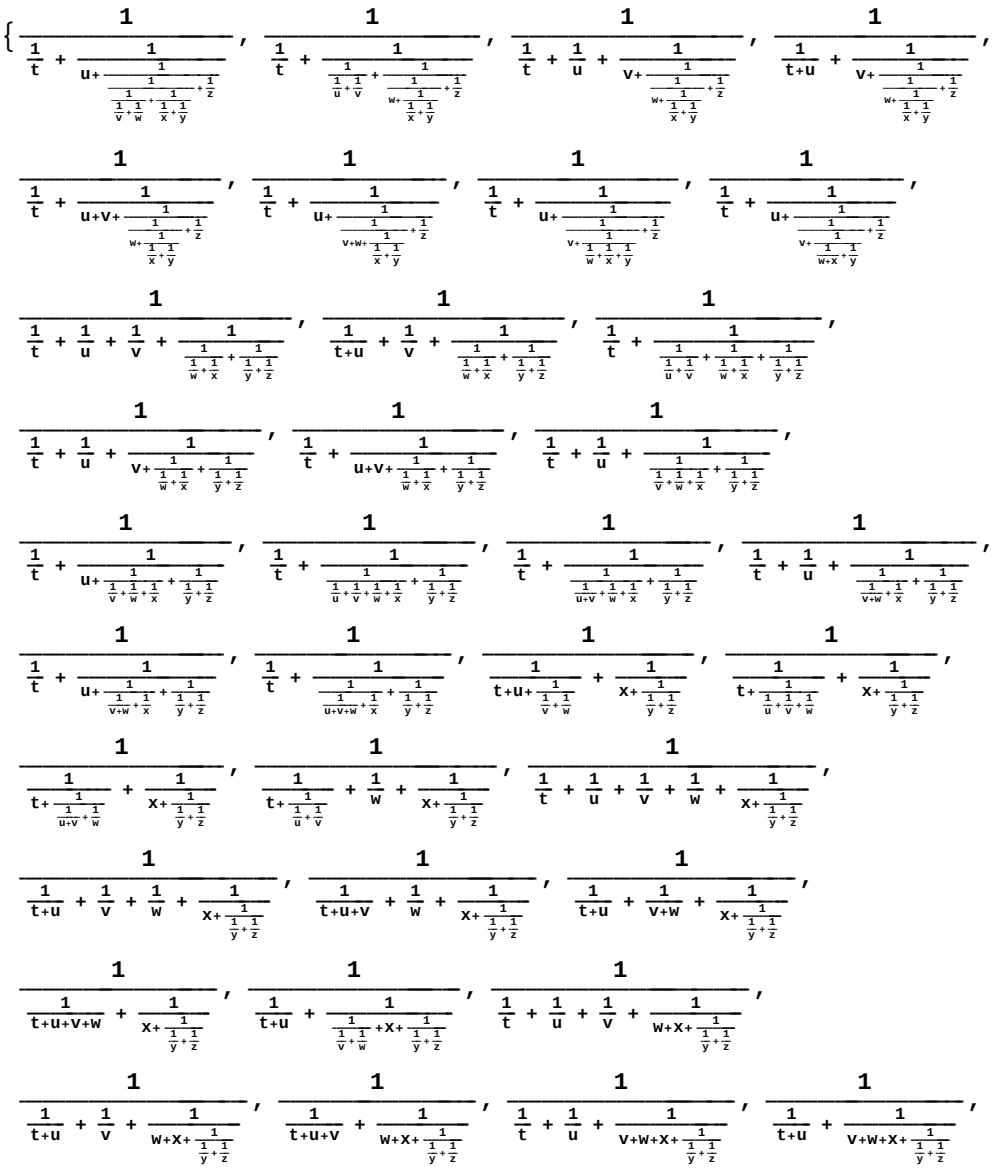
Out[203]=
```
{1, 2, 4, 10, 24, 66}
```

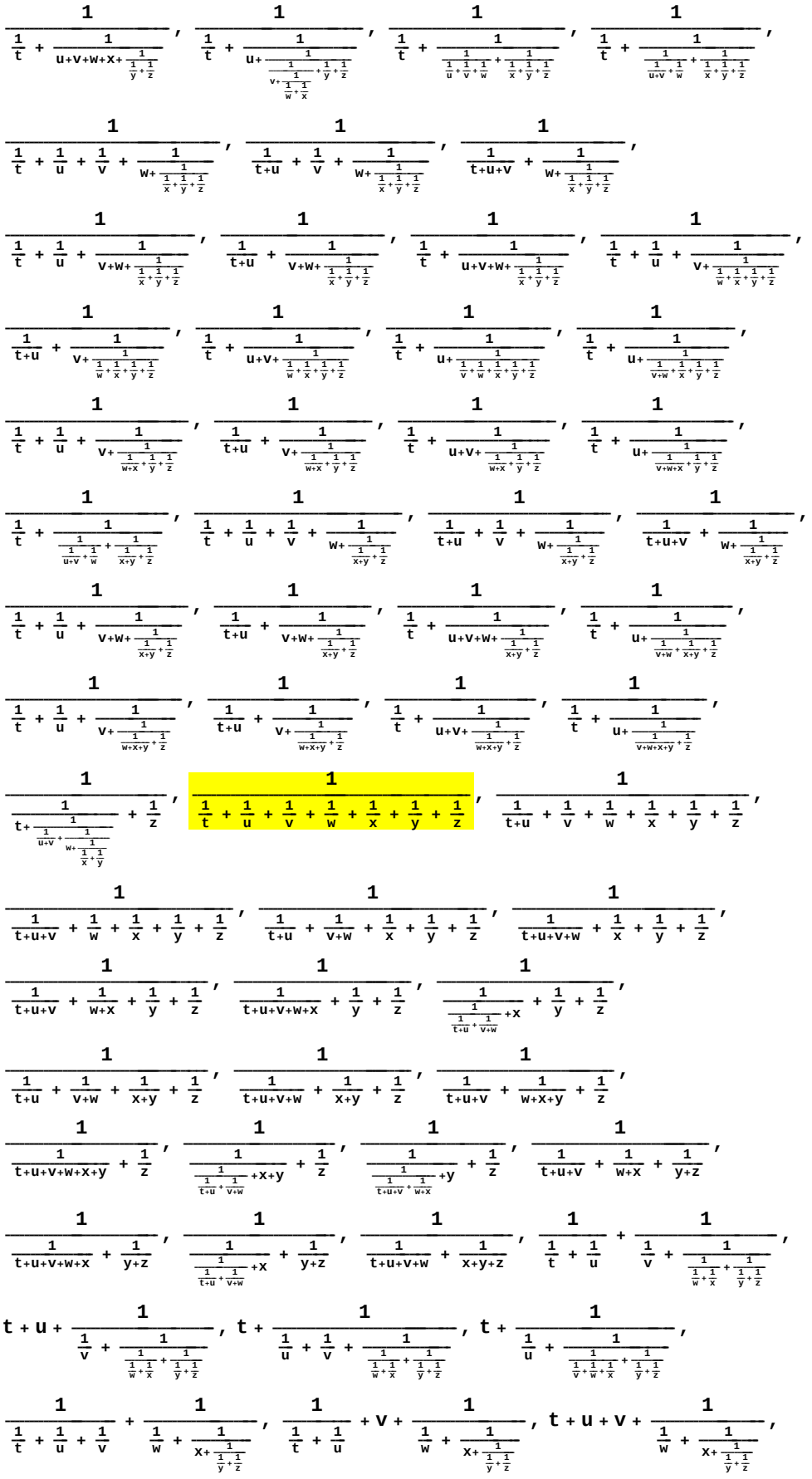
… there are supposed to be 66! The question is, what do the 2 missing circuits look like, and why does our representation scheme somehow not encompass these 2 circuits. The running time for this output clearly indicates the beginning of the (inevitable?) combinatorial explosion.

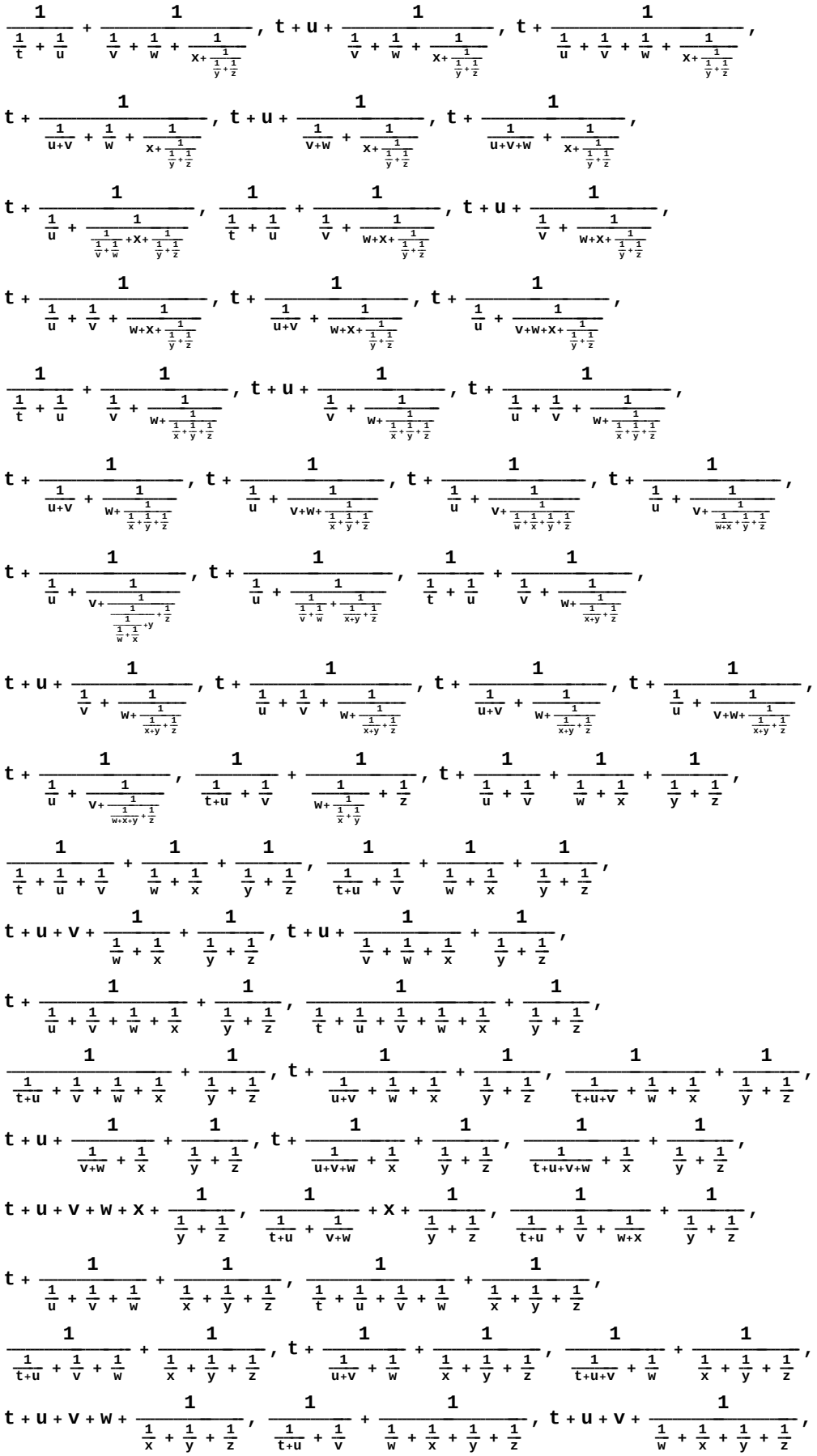
■ Section 15 - What about 7 Resistors, Then? Another Surprise?

Again the results are somewhat surprising, but now not unexpected. Here explicitly are all the 7-element circuits found by the our approach:

In[204]:=
```
circuitsWith7Elements =
```







$$t+u+\frac{1}{\frac{1}{v}+\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},\ t+\frac{1}{\frac{1}{u}+\frac{1}{v}+\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},$$

$$t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},\ t+u+\frac{1}{\frac{1}{v+w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},\ t+\frac{1}{\frac{1}{u+v+w}+\frac{1}{x}+\frac{1}{y}+\frac{1}{z}},$$

$$\frac{1}{\frac{1}{t+u}+\frac{1}{v}}+\frac{1}{\frac{1}{w+x}+\frac{1}{y}+\frac{1}{z}},\ t+u+v+\frac{1}{\frac{1}{w+x}+\frac{1}{y}+\frac{1}{z}},\ t+\frac{1}{\frac{1}{u+v}+\frac{1}{w+x}+\frac{1}{y}+\frac{1}{z}},$$

$$t+u+\frac{1}{\frac{1}{v+w+x}+\frac{1}{y}+\frac{1}{z}},\ t+\frac{1}{\frac{1}{u+v+w+x}+\frac{1}{y}+\frac{1}{z}},\ t+\frac{1}{\frac{1}{u+v}+\frac{1}{w}}+\frac{1}{\frac{1}{x+y}+\frac{1}{z}},$$

$$\frac{1}{\frac{1}{t+u+v}+\frac{1}{w}}+\frac{1}{\frac{1}{x+y}+\frac{1}{z}},\ t+u+v+w+\frac{1}{\frac{1}{x+y}+\frac{1}{z}},\ t+u+\frac{1}{\frac{1}{v+w}+\frac{1}{x+y}+\frac{1}{z}},$$

$$t+\frac{1}{\frac{1}{u+v+w}+\frac{1}{x+y}+\frac{1}{z}},\ t+u+v+\frac{1}{\frac{1}{w+x+y}+\frac{1}{z}},\ t+u+\frac{1}{\frac{1}{v+w+x+y}+\frac{1}{z}},$$

$$t+\frac{1}{\frac{1}{u+v+w+x+y}+\frac{1}{z}},\ t+\frac{1}{\frac{1}{\frac{1}{u+v}+\frac{1}{w+x}}+y}+\frac{1}{z},\ \frac{1}{\frac{1}{t+\frac{1}{\frac{1}{u}+\frac{1}{v}}}+\frac{1}{w+\frac{1}{\frac{1}{x}+\frac{1}{y}}}}}+z,$$

$$t+u+v+w+x+y+z,\ \frac{1}{\frac{1}{t+u}+\frac{1}{v+w}}+x+y+z,\ \frac{1}{\frac{1}{t+u+v}+\frac{1}{w+x}}+y+z,$$

$$\frac{1}{\frac{1}{t+u}+\frac{1}{v+w}+\frac{1}{x+y}}+z,\ \frac{1}{\frac{1}{t+u+v+w}+\frac{1}{x+y}}+z,\ \frac{1}{\frac{1}{t+u+v}+\frac{1}{w+x+y}}+z\Big\};$$

In[205]:=
```
Dimensions @ circuitsWith7Elements
```

Out[205]=
```
{170}
```

We found 170 circuits, but …

In[206]:=
```
seriesParallelNetworks[7]
```

Out[206]=
```
{1, 2, 4, 10, 24, 66, 180}
```

… there are supposed to be 180! Now there are 10 missing circuits. (The running time for this output was 22.3 hours on an AMD64 2 GHz XP Pro machine.)

■ Section 16 - Discussion and Conclusion.

What could be wrong? It might be that our nesting representation system cannot represent the "missing" circuits. It might just be that we do not generate the nesting representation for the missing circuits in our construction process, and that a somewhat more complex process is needed to produce the apparently more intricate nestings required. It might even be that the missing circuits *are* present in the candidate list **`expressionListT1T2`**, but are collapsed into other circuits during the **`Union[Sort[(Sort[#])& /@ expressionListT1T2]`**, **`SameTest→ equivalentStructuresQ]`**
process. But why does the method work for n = 2, 3, 4, and 5 elements, but then not for 6, 7, and presumably larger values? (It is at least the case that in the sets of 64 and 170 respectively, half of the circuits are essentially series and half are essentially parallel.) At the time of writing the reason for the missing circuits is an open question.

Given the very rapid proliferation of possible series-parallel circuits as the number of circuit elements is increased, the missing circuits might never be important in terms of the pragmatic aim of this work. Substituting 4 prime-valued resistors (2 Ω, 3 Ω, 5 Ω, and 7 Ω) into all possible positions in the 10 4-resistor circuits provides 48 distinct total resistance values in the feasible range from the all-in-parallel circuit (0.850202 Ω) to the all-in-series circuit (17 Ω). In the lower end of this range, where the total resistances are distributed more densely, there are 4 available values between 3.5 Ω and 4.5 Ω. Substituting 5 prime-valued resistors (2 Ω, 3 Ω, 5 Ω, 7 Ω, and 11 Ω) into all possible positions in the 24 5-resistor circuits results in 453 values between the all-in-parallel circuit (0.789204 Ω) and the all-in-series circuit (28 Ω), with 11 different values between 11 Ω and 12 Ω.

It appears that we can provide sufficient options to construct a circuit with total resistance close to any particular value in the feasible range for a given set of resistors, so the missing circuits are probably not important from a pragmatic point of view. They are, however, a matter for further investigation. The situation is intriguing.

At least in every case investigated, the all-series and the all-parallel circuits are found, along with many that appear to be as "deeply" structured as might be thought possible, such as

$$u + \cfrac{1}{\frac{1}{v} + \frac{1}{w} + \cfrac{1}{x + \cfrac{1}{\frac{1}{y} + \frac{1}{z}}}}$$

with 6 resistors, or

$$\cfrac{1}{t + \cfrac{1}{\cfrac{1}{u + v} + \cfrac{1}{w + \cfrac{1}{\frac{1}{x} + \frac{1}{y}}}} + \frac{1}{z}}$$

with 7 resistors.

What do the missing circuits look like?

Why can't we exhibit them?

Is the problem in the mathematics or the *Mathematica?*