Calculating the resistance between two points in an arbitrary finite network of resistors

Paul Breeuwsma

18 September 2022

Contents

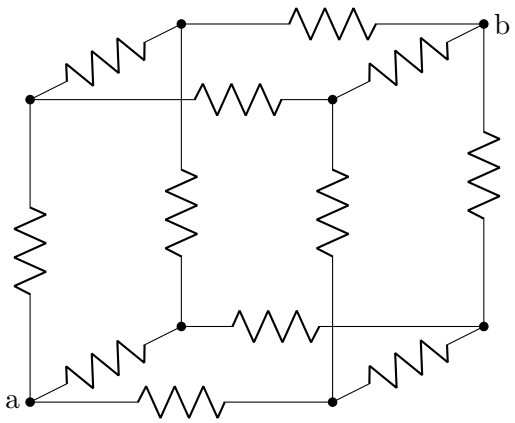
1 Introduction **1**
1.1 Examples 2

2 A general method **3**
2.1 Calculating all pairs 5

3 Automating using Wolfram Mathematica **6**
3.1 The code 6
3.2 The examples 7
3.3 A fancy example 8
3.4 Resistance matrix 9

1 Introduction

When combining multiple resistors in series, in parallel, or in some other way, this combination of resistors behaves as if it is a single resistor. The resistance of this combination of resistors can be calculated. In this article I will explain how. The article is written from a theoretical/mathematical point of view. All resistors are assumed to be ideal resistors. So, for example, the maximum power dissipation of a resistor is not taken into account.

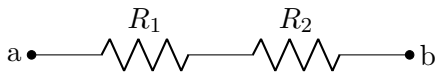


$R'_{\mathrm{ab}} = ?$

1.1 Examples

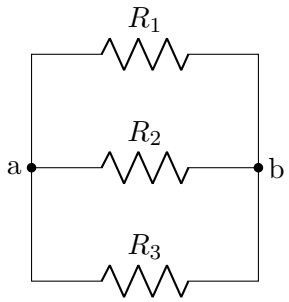
In this article R is used for the resistance of a single resistor, while R' is used for the equivalent resistance of a combination of resistors.

If you put resistors in series, their resistance adds up.



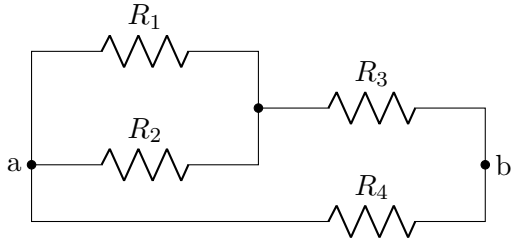
$$R'_{\mathrm{ab}} = R_1 + R_2 \tag{1}$$

If you put resistors in parallel, their conductance (one divided by the resistance) adds up. For example in the case of three parallel resistors:



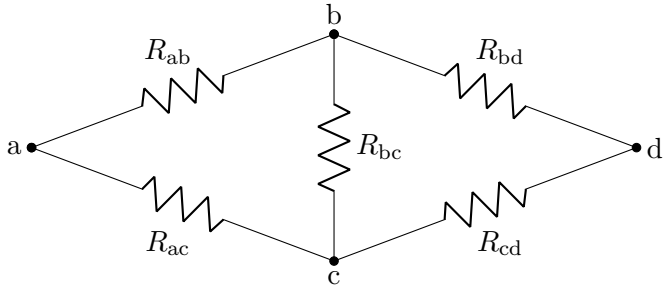
$$R'_{\mathrm{ab}} = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3}} \tag{2}$$

By combining the rules for calculating resistors in series and in parallel, many networks of resistors can be calculated by applying the rules one at a time. In the example below, start by replacing R_1 and R_2 by a resistor of equivalent resistance. This new resistor is now in series with R_3, which allows you to do the next step.



$$R'_{\mathrm{ab}} = \frac{1}{\frac{1}{\frac{1}{\frac{1}{R_1} + \frac{1}{R_2}} + R_3} + \frac{1}{R_4}} \tag{3}$$

Now let's look at the resistor network below. Using the same method you should be able to calculate the resistance between any two points in this network, except for between a and d.



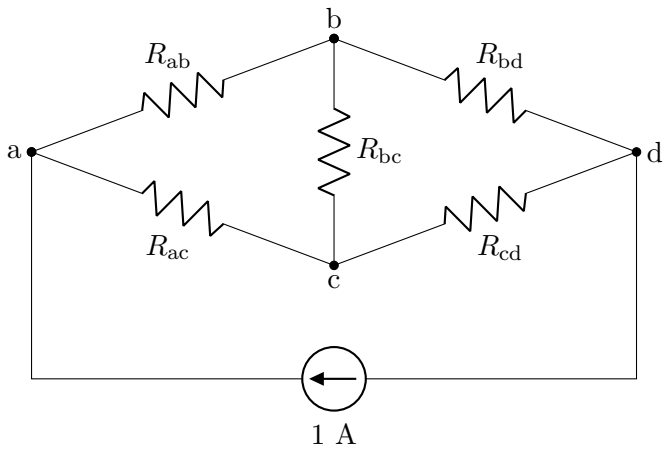
$$R'_{\mathrm{ad}} = ? \tag{4}$$

For calculating the resistance between a and d, a different approach is needed, which will be shown in the next section.

2 A general method

In this section I will show how to calculate the equivalent resistance between a and d in example (4), but you can use the same method for any finite network of resistors.

First we add a constant current source to the circuit.



We are now going to calculate the voltages at the nodes of this circuit.

According to Kirchhoff's first law, every node must have the same amount of current flowing in as flowing out. Let x and y be arbitrary nodes in the network and let I_{xy} be the amount of current in ampere that flows directly from node x to node y. In general we have $I_{xy} = -I_{yx}$. For nodes a, b, c and d, we have:

$$\begin{aligned} I_{\mathrm{ab}} + I_{\mathrm{ac}} &= 1 \\ I_{\mathrm{ba}} + I_{\mathrm{bc}} + I_{\mathrm{bd}} &= 0 \\ I_{\mathrm{ca}} + I_{\mathrm{cb}} + I_{\mathrm{cd}} &= 0 \\ I_{\mathrm{db}} + I_{\mathrm{dc}} &= -1 \end{aligned} \tag{5}$$

Let V_x be the voltage at node x. Ohm's law states:

$$I_{xy} = \frac{V_x - V_y}{R_{xy}} \tag{6}$$

We can use this formula to rewrite (5). We also use $R_{xy} = R_{yx}$.

$$\begin{aligned} \frac{V_{\mathrm{a}} - V_{\mathrm{b}}}{R_{\mathrm{ab}}} + \frac{V_{\mathrm{a}} - V_{\mathrm{c}}}{R_{\mathrm{ac}}} &= 1 \\ \frac{V_{\mathrm{b}} - V_{\mathrm{a}}}{R_{\mathrm{ab}}} + \frac{V_{\mathrm{b}} - V_{\mathrm{c}}}{R_{\mathrm{bc}}} + \frac{V_{\mathrm{b}} - V_{\mathrm{d}}}{R_{\mathrm{bd}}} &= 0 \\ \frac{V_{\mathrm{c}} - V_{\mathrm{a}}}{R_{\mathrm{ac}}} + \frac{V_{\mathrm{c}} - V_{\mathrm{b}}}{R_{\mathrm{bc}}} + \frac{V_{\mathrm{c}} - V_{\mathrm{d}}}{R_{\mathrm{cd}}} &= 0 \\ \frac{V_{\mathrm{d}} - V_{\mathrm{b}}}{R_{\mathrm{bd}}} + \frac{V_{\mathrm{d}} - V_{\mathrm{c}}}{R_{\mathrm{cd}}} &= -1 \end{aligned}$$

To make this a bit more readable, we are going to use the conductance G instead of the resistance R.

$$G_{xy} = \frac{1}{R_{xy}}$$

If we reorder the elements and factor out the voltages, the equations become:

$$\begin{array}{rcrcrcrcl}
(G_{\mathrm{ab}}+G_{\mathrm{ac}})V_{\mathrm{a}} & + & (-G_{\mathrm{ab}})V_{\mathrm{b}} & + & (-G_{\mathrm{ac}})V_{\mathrm{c}} & & & = & 1 \\
(-G_{\mathrm{ab}})V_{\mathrm{a}} & + & (G_{\mathrm{ab}}+G_{\mathrm{bc}}+G_{\mathrm{bd}})V_{\mathrm{b}} & + & (-G_{\mathrm{bc}})V_{\mathrm{c}} & + & (-G_{\mathrm{bd}})V_{\mathrm{d}} & = & 0 \\
(-G_{\mathrm{ac}})V_{\mathrm{a}} & + & (-G_{\mathrm{bc}})V_{\mathrm{b}} & + & (G_{\mathrm{ac}}+G_{\mathrm{bc}}+G_{\mathrm{cd}})V_{\mathrm{c}} & + & (-G_{\mathrm{cd}})V_{\mathrm{d}} & = & 0 \\
 & & (-G_{\mathrm{bd}})V_{\mathrm{b}} & + & (-G_{\mathrm{cd}})V_{\mathrm{c}} & + & (G_{\mathrm{bd}}+G_{\mathrm{cd}})V_{\mathrm{d}} & = & -1
\end{array}$$

This system of linear equations can be written as one matrix equation.

$$\begin{bmatrix}
G_{\mathrm{ab}}+G_{\mathrm{ac}} & -G_{\mathrm{ab}} & -G_{\mathrm{ac}} & 0 \\
-G_{\mathrm{ab}} & G_{\mathrm{ab}}+G_{\mathrm{bc}}+G_{\mathrm{bd}} & -G_{\mathrm{bc}} & -G_{\mathrm{bd}} \\
-G_{\mathrm{ac}} & -G_{\mathrm{bc}} & G_{\mathrm{ac}}+G_{\mathrm{bc}}+G_{\mathrm{cd}} & -G_{\mathrm{cd}} \\
0 & -G_{\mathrm{bd}} & -G_{\mathrm{cd}} & G_{\mathrm{bd}}+G_{\mathrm{cd}}
\end{bmatrix}
\begin{bmatrix} V_{\mathrm{a}} \\ V_{\mathrm{b}} \\ V_{\mathrm{c}} \\ V_{\mathrm{d}} \end{bmatrix}
=
\begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix} \tag{7}$$

All values of G are known and we want to find the voltages. Now this equation does not have a unique solution. This is because voltages are relative. If you have a solution, then you can add some number to all voltages, and it will also be a valid solution. To obtain a unique solution, we set the node that is connected to the negative side of the power supply to 0 V.

$$V_{\mathrm{d}} = 0$$

This also simplifies the equation, since we can drop the column associated with V_{d} from the matrix.

$$\begin{bmatrix}
G_{\mathrm{ab}}+G_{\mathrm{ac}} & -G_{\mathrm{ab}} & -G_{\mathrm{ac}} \\
-G_{\mathrm{ab}} & G_{\mathrm{ab}}+G_{\mathrm{bc}}+G_{\mathrm{bd}} & -G_{\mathrm{bc}} \\
-G_{\mathrm{ac}} & -G_{\mathrm{bc}} & G_{\mathrm{ac}}+G_{\mathrm{bc}}+G_{\mathrm{cd}} \\
0 & -G_{\mathrm{bd}} & -G_{\mathrm{cd}}
\end{bmatrix}
\begin{bmatrix} V_{\mathrm{a}} \\ V_{\mathrm{b}} \\ V_{\mathrm{c}} \end{bmatrix}
=
\begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix} \tag{8}$$

We can simplify this equation even more. Because (8) has more equations than unknowns, it is an overdetermined system. Every row is the negation of the sum of the other rows, so every row is a linear combination of the other rows. This means we can remove an arbitrary row and still have the same solution. We delete the row associated with d.

$$\begin{bmatrix}
G_{\mathrm{ab}}+G_{\mathrm{ac}} & -G_{\mathrm{ab}} & -G_{\mathrm{ac}} \\
-G_{\mathrm{ab}} & G_{\mathrm{ab}}+G_{\mathrm{bc}}+G_{\mathrm{bd}} & -G_{\mathrm{bc}} \\
-G_{\mathrm{ac}} & -G_{\mathrm{bc}} & G_{\mathrm{ac}}+G_{\mathrm{bc}}+G_{\mathrm{cd}}
\end{bmatrix}
\begin{bmatrix} V_{\mathrm{a}} \\ V_{\mathrm{b}} \\ V_{\mathrm{c}} \end{bmatrix}
=
\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{9}$$

The solution to this equation can be found using an algorithm called Gaussian elimination. This can be done by hand, but I prefer to let Mathematica do the work.

```
m = {{1/rab + 1/rac, -1/rab, -1/rac},
     {-1/rab, 1/rab + 1/rbc + 1/rbd, -1/rbc},
     {-1/rac, -1/rbc, 1/rac + 1/rbc + 1/rcd}};
i = {1, 0, 0};
v = LinearSolve[m, i] // Simplify
```

The vector V will have the value:

$$\begin{bmatrix} V_{\mathrm{a}} \\ V_{\mathrm{b}} \\ V_{\mathrm{c}} \end{bmatrix} =
\begin{bmatrix}
\dfrac{R_{\mathrm{bc}}R_{\mathrm{bd}}R_{\mathrm{cd}} + R_{\mathrm{ab}}(R_{\mathrm{bc}}+R_{\mathrm{bd}})R_{\mathrm{cd}} + R_{\mathrm{ac}}R_{\mathrm{bd}}(R_{\mathrm{bc}}+R_{\mathrm{cd}}) + R_{\mathrm{ab}}R_{\mathrm{ac}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}+R_{\mathrm{cd}})}{R_{\mathrm{bc}}(R_{\mathrm{bd}}+R_{\mathrm{cd}}) + R_{\mathrm{ab}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}+R_{\mathrm{cd}}) + R_{\mathrm{ac}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}+R_{\mathrm{cd}})} \\
\dfrac{R_{\mathrm{bd}}((R_{\mathrm{ab}}+R_{\mathrm{bc}})R_{\mathrm{cd}} + R_{\mathrm{ac}}(R_{\mathrm{bc}}+R_{\mathrm{cd}}))}{R_{\mathrm{bc}}(R_{\mathrm{bd}}+R_{\mathrm{cd}}) + R_{\mathrm{ab}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}+R_{\mathrm{cd}}) + R_{\mathrm{ac}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}+R_{\mathrm{cd}})} \\
\dfrac{((R_{\mathrm{ac}}+R_{\mathrm{bc}})R_{\mathrm{bd}} + R_{\mathrm{ab}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}))R_{\mathrm{cd}}}{R_{\mathrm{bc}}(R_{\mathrm{bd}}+R_{\mathrm{cd}}) + R_{\mathrm{ab}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}+R_{\mathrm{cd}}) + R_{\mathrm{ac}}(R_{\mathrm{bc}}+R_{\mathrm{bd}}+R_{\mathrm{cd}})}
\end{bmatrix}$$

So now we know the voltage at every node in the resistor network. But we want to know the equivalent resistance between a and d. We know the voltages at a and d, and we know that a current of 1 A flows. So we can use Ohm's law (6) again to calculate the resistance.

$$R'_{\mathrm{ad}} = \frac{V_{\mathrm{a}} - V_{\mathrm{d}}}{I'_{\mathrm{ad}}} = V_{\mathrm{a}}$$

2.1 Calculating all pairs

Let's say we do not want the find the resistance between two specific nodes in a resistor network, but calculate the resistance between any pair of nodes. Of course we can simply repeat the process for different combinations of nodes, but there is a more efficient way.

Let's take a look at (9) again. If instead of finding the resistance between a and d, we want to find the resistance between b and d or between c and d, then the only thing that changes in this equation is the position of the 1. We can combine these three equations to one equation.

Let V_{xy} be the voltage at node y, when the positive side of the power supply is connected to node x and the negative side to node d. To get these voltages, we have to solve:

$$\begin{bmatrix} G_{\mathrm{ab}} + G_{\mathrm{ac}} & -G_{\mathrm{ab}} & -G_{\mathrm{ac}} \\ -G_{\mathrm{ab}} & G_{\mathrm{ab}} + G_{\mathrm{bc}} + G_{\mathrm{bd}} & -G_{\mathrm{bc}} \\ -G_{\mathrm{ac}} & -G_{\mathrm{bc}} & G_{\mathrm{ac}} + G_{\mathrm{bc}} + G_{\mathrm{cd}} \end{bmatrix} \begin{bmatrix} V_{\mathrm{aa}} & V_{\mathrm{ba}} & V_{\mathrm{ca}} \\ V_{\mathrm{ab}} & V_{\mathrm{bb}} & V_{\mathrm{cb}} \\ V_{\mathrm{ac}} & V_{\mathrm{bc}} & V_{\mathrm{cc}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

So, we have to calculate the inverse of the G matrix to get the V matrix.

Since we know $R'_{x\mathrm{d}} = V_{xx}$, we can look at the diagonal of the V matrix to get the resistances.

To get all pairs, we have to repeat for $k \in \{1, 2, 3, 4\}$:

1. Remove the kth row and the kth column from the matrix in (7).
2. Calculate the matrix inverse
3. Take the items at the diagonal

See section 3.4 for an implementation.

3 Automating using Wolfram Mathematica

In the previous section we used Mathematica to solve the final matrix equation. In this section the whole process will be automated.

3.1 The code

The two functions below get a graph object and two vertices of that graph as input. Every edge of the graph represents a resistor. The resistor values are stored as edge weights. If no edge weight is specified, a resistance of 1 Ω is assumed.

```
voltagesOneAmp[graph_, vertexIn_, vertexOut_] := Module[
    {g = graph, indexIn, indexOut, m, i}
    ,
    indexIn = VertexIndex[g, vertexIn];
    indexOut = VertexIndex[g, vertexOut];
    If[WeightedGraphQ[g],
        AnnotationValue[g, EdgeWeight] = 1 / AnnotationValue[g, EdgeWeight]
    ];
    m = WeightedAdjacencyMatrix[g];
    m = DiagonalMatrix[Total[m]] - m;
    m = Drop[m, {indexOut}, {indexOut}];
    i = Drop[SparseArray[{indexIn -> 1}, VertexCount[g]], {indexOut}];
    Insert[LinearSolve[m, i], 0, indexOut]
];

equivalentResistance[graph_, vertex1_, vertex2_] :=
    voltagesOneAmp[graph, vertex1, vertex2][[VertexIndex[graph, vertex1]]];
```

In `voltagesOneAmp`, equation (9) is made by generating the matrix `m` and the right-hand vector `i`, and then solved using the `LinearSolve` command. It returns the voltages at the vertices, when a 1 A power supply is connected between the two specified vertices.

Let's execute this code on example (4). First it changes the edge weights by replacing resistance with conductance (1/resistance). Later the line "`m = DiagonalMatrix[Total[m]] - m;`" will calculate the matrix in (7) like this:

$$\underset{\texttt{DiagonalMatrix[Total[m]]}}{\begin{bmatrix} G_{ab}+G_{ac} & 0 & 0 & 0 \\ 0 & G_{ab}+G_{bc}+G_{bd} & 0 & 0 \\ 0 & 0 & G_{ac}+G_{bc}+G_{cd} & 0 \\ 0 & 0 & 0 & G_{bd}+G_{cd} \end{bmatrix}} - \underset{\texttt{m}}{\begin{bmatrix} 0 & G_{ab} & G_{ac} & 0 \\ G_{ab} & 0 & G_{bc} & G_{bd} \\ G_{ac} & G_{bc} & 0 & G_{cd} \\ 0 & G_{bd} & G_{cd} & 0 \end{bmatrix}}$$

The line after that drops the row and column associated with d from `m` to arrive at the matrix in (9).

Add some `Print[]` statements in the functions to have a closer look at what's happening.

3.2 The examples

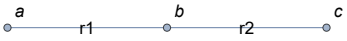
We will try the code on the four examples from the introduction. The following function will be used to show a graph and a formula for the resistance.

```
showExample[edges_, weights_, from_, to_] := Module[{e, g, r},
    e = UndirectedEdge @@ # & /@ edges;
    g = EdgeTaggedGraph[e, EdgeWeight -> weights, EdgeLabels -> "EdgeWeight",
                        VertexLabels -> Automatic, ImageSize -> Small];
    r = equivalentResistance[g, from, to] // Simplify;
    Print[g];
    Print[r];
];
```

Note that the formulas we get here for example (2) and (3) are different from the formulas we found in the introduction, but they are only different representations of the same function. In other words, they'll give the same value.

Example (1):

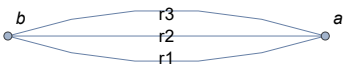
```
showExample[{a.b, b.c}, {r1, r2}, a, c];
```



$$R_1 + R_2$$

Example (2):

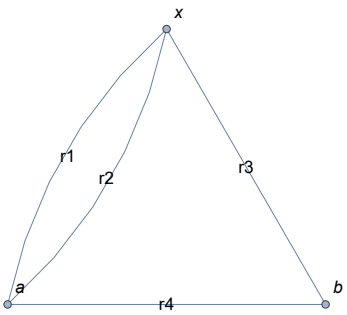
```
showExample[{a.b, a.b, a.b}, {r1, r2, r3}, a, b];
```



$$\frac{R_1 R_2 R_3}{R_2 R_3 + R_1(R_2 + R_3)}$$

Example (3):

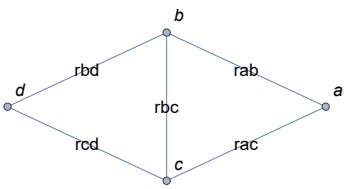
```
showExample[{a.x, a.x, x.b, a.b}, {r1, r2, r3, r4}, a, b];
```



$$\frac{(R_2 R_3 + R_1(R_2 + R_3))R_4}{R_2(R_3 + R_4) + R_1(R_2 + R_3 + R_4)}$$

Example (4):

```
showExample[{a.b, a.c, b.c, b.d, c.d}, {rab, rac, rbc, rbd, rcd}, a, d];
```



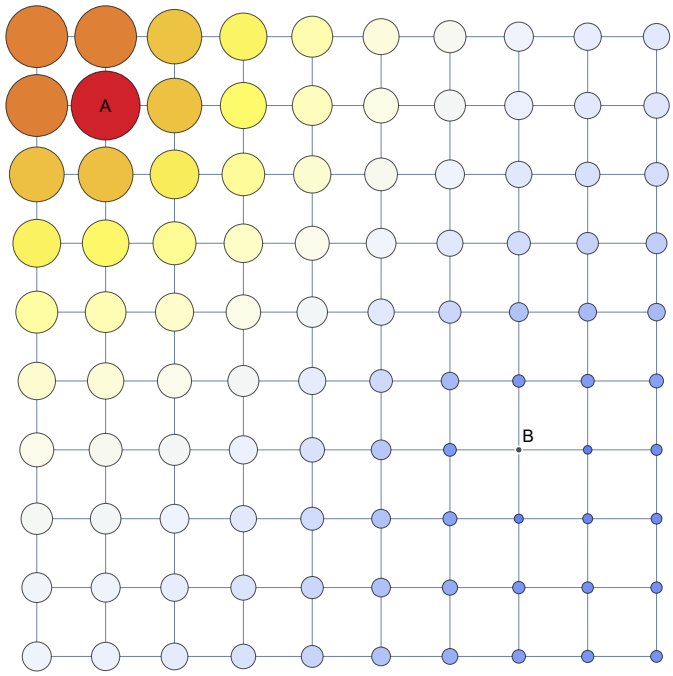
$$\frac{R_{bc}R_{bd}R_{cd} + R_{ab}(R_{bc} + R_{bd})R_{cd} + R_{ac}R_{bd}(R_{bc} + R_{cd}) + R_{ab}R_{ac}(R_{bc} + R_{bd} + R_{cd})}{R_{bc}(R_{bd} + R_{cd}) + R_{ab}(R_{bc} + R_{bd} + R_{cd}) + R_{ac}(R_{bc} + R_{bd} + R_{cd})}$$

3.3 A fancy example

On the webpage `https://rosettacode.org/wiki/Resistor_mesh` it is asked to calculate the resistance between two points on a 10x10 grid of nodes connected by 1 Ω resistors.

We can do that, and also make a nice picture. The resistor network is shown below, with our 1 ampere current source connected between A and B. The radius of every node is proportional to its voltage, and the color is also a function of the voltage.

```
a = 19;
b = 74;
g = GridGraph[{10, 10}];
v = voltagesOneAmp[g, a, b];
r = v[[a]];
colors = ColorData["TemperatureMap"];
Graph[g,
    VertexLabels -> {a -> Placed["A", Center], b -> "B"},
    VertexSize -> MapThread[#1 -> #2/r &, {VertexList[g], v}],
    VertexStyle -> MapThread[#1 -> colors[#2/r] &, {VertexList[g], v}]
]
TildeTilde[r, N[r]]
```



$$\frac{45585913702572}{283319837425200} \approx 1.60899\ \Omega$$

Similarly, the resistance between two points in a resistor network in the form of a cube, as shown on page 1, can be found with help of the `HypercubeGraph` function.

3.4 Resistance matrix

The following function calculates the resistance matrix of a resistor network. That is, a matrix R' such that the element at row x and column y, R'_{xy}, is the resistance between node x and y. The function is made by editing the `voltagesOneAmp` function to solve equation (10) once for every node.

```
resistanceMatrix[graph_] := Module[{g = graph, m},
    If[WeightedGraphQ[g],
        AnnotationValue[g, EdgeWeight] = 1 / AnnotationValue[g, EdgeWeight]
    ];
    m = WeightedAdjacencyMatrix[g];
    m = DiagonalMatrix[Total[m]] - m;
    Table[
        Insert[Drop[m, {i}, {i}] // Inverse // Diagonal, 0, i]
        ,
        {i, VertexCount[g]}
    ]
];
```