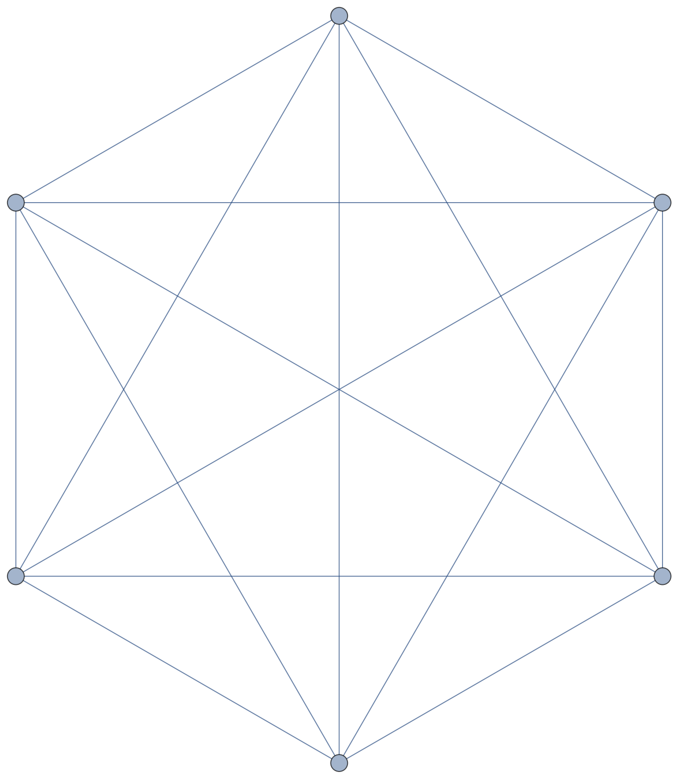
Reciprocal Multifactorial Constants

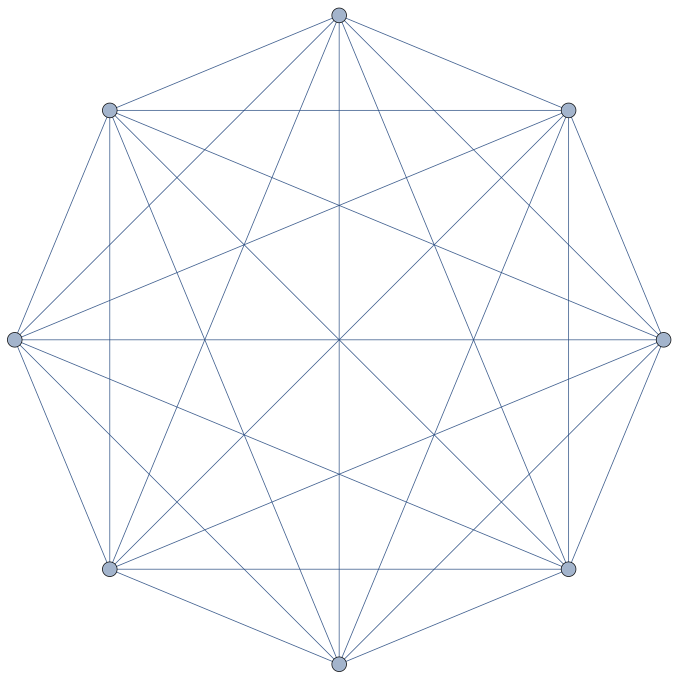
Perfect matchings for K_ 6 and K_8

In[ ]:=
```
k6 = CompleteGraph[6]
k8 = CompleteGraph[8]
```

Out[ ]=



Out[ ]=



In[ ]:=
```
l6 = Length[FindIndependentEdgeSet[k6]]
l8 = Length[FindIndependentEdgeSet[k8]]
```

Out[ ]= 3

Out[ ]= 4

In[●]:=
```
esl6 = Select[Subsets[EdgeList[k6], {l6}], IndependentEdgeSetQ[k6, #] &]
esl8 = Select[Subsets[EdgeList[k8], {l8}], IndependentEdgeSetQ[k8, #] &]
```

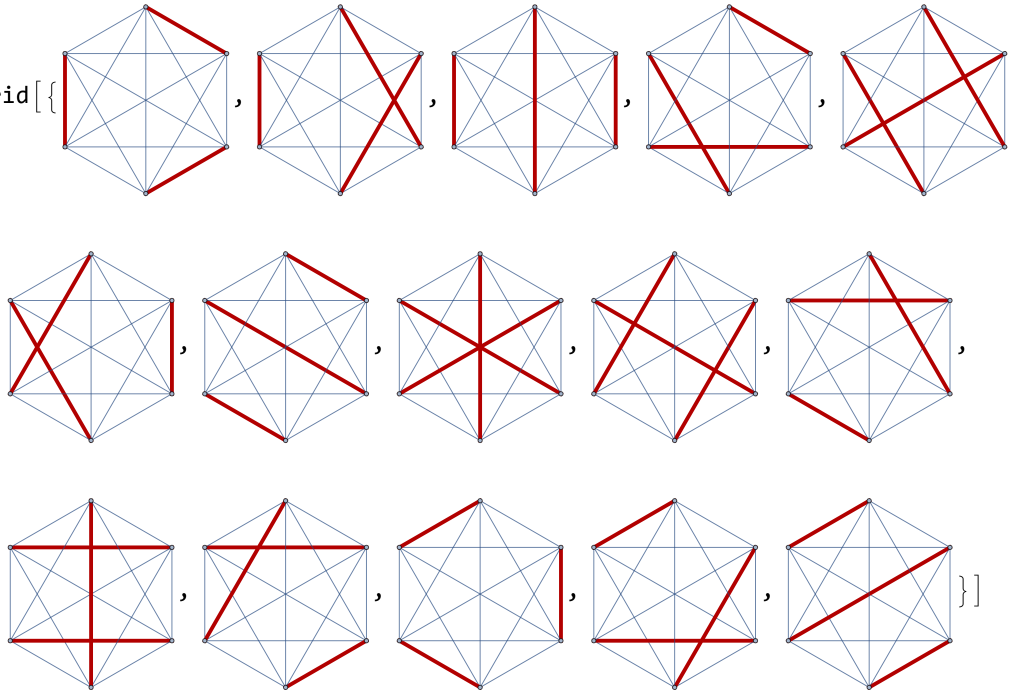
Out[●]=
```
{{1↔2, 3↔4, 5↔6}, {1↔2, 3↔5, 4↔6}, {1↔2, 3↔6, 4↔5}, {1↔3, 2↔4, 5↔6},
 {1↔3, 2↔5, 4↔6}, {1↔3, 2↔6, 4↔5}, {1↔4, 2↔3, 5↔6}, {1↔4, 2↔5, 3↔6},
 {1↔4, 2↔6, 3↔5}, {1↔5, 2↔3, 4↔6}, {1↔5, 2↔4, 3↔6}, {1↔5, 2↔6, 3↔4},
 {1↔6, 2↔3, 4↔5}, {1↔6, 2↔4, 3↔5}, {1↔6, 2↔5, 3↔4}}
```

Out[●]=
```
{{1↔2, 3↔4, 5↔6, 7↔8}, {1↔2, 3↔4, 5↔7, 6↔8}, {1↔2, 3↔4, 5↔8, 6↔7},
 {1↔2, 3↔5, 4↔6, 7↔8}, {1↔2, 3↔5, 4↔7, 6↔8}, {1↔2, 3↔5, 4↔8, 6↔7},
 {1↔2, 3↔6, 4↔5, 7↔8}, {1↔2, 3↔6, 4↔7, 5↔8}, {1↔2, 3↔6, 4↔8, 5↔7},
 {1↔2, 3↔7, 4↔5, 6↔8}, {1↔2, 3↔7, 4↔6, 5↔8}, {1↔2, 3↔7, 4↔8, 5↔6},
 {1↔2, 3↔8, 4↔5, 6↔7}, {1↔2, 3↔8, 4↔6, 5↔7}, {1↔2, 3↔8, 4↔7, 5↔6},
 {1↔3, 2↔4, 5↔6, 7↔8}, {1↔3, 2↔4, 5↔7, 6↔8}, {1↔3, 2↔4, 5↔8, 6↔7},
 {1↔3, 2↔5, 4↔6, 7↔8}, {1↔3, 2↔5, 4↔7, 6↔8}, {1↔3, 2↔5, 4↔8, 6↔7},
 {1↔3, 2↔6, 4↔5, 7↔8}, {1↔3, 2↔6, 4↔7, 5↔8}, {1↔3, 2↔6, 4↔8, 5↔7},
 {1↔3, 2↔7, 4↔5, 6↔8}, {1↔3, 2↔7, 4↔6, 5↔8}, {1↔3, 2↔7, 4↔8, 5↔6},
 {1↔3, 2↔8, 4↔5, 6↔7}, {1↔3, 2↔8, 4↔6, 5↔7}, {1↔3, 2↔8, 4↔7, 5↔6},
 {1↔4, 2↔3, 5↔6, 7↔8}, {1↔4, 2↔3, 5↔7, 6↔8}, {1↔4, 2↔3, 5↔8, 6↔7},
 {1↔4, 2↔5, 3↔6, 7↔8}, {1↔4, 2↔5, 3↔7, 6↔8}, {1↔4, 2↔5, 3↔8, 6↔7},
 {1↔4, 2↔6, 3↔5, 7↔8}, {1↔4, 2↔6, 3↔7, 5↔8}, {1↔4, 2↔6, 3↔8, 5↔7},
 {1↔4, 2↔7, 3↔5, 6↔8}, {1↔4, 2↔7, 3↔6, 5↔8}, {1↔4, 2↔7, 3↔8, 5↔6},
 {1↔4, 2↔8, 3↔5, 6↔7}, {1↔4, 2↔8, 3↔6, 5↔7}, {1↔4, 2↔8, 3↔7, 5↔6},
 {1↔5, 2↔3, 4↔6, 7↔8}, {1↔5, 2↔3, 4↔7, 6↔8}, {1↔5, 2↔3, 4↔8, 6↔7},
 {1↔5, 2↔4, 3↔6, 7↔8}, {1↔5, 2↔4, 3↔7, 6↔8}, {1↔5, 2↔4, 3↔8, 6↔7},
 {1↔5, 2↔6, 3↔4, 7↔8}, {1↔5, 2↔6, 3↔7, 4↔8}, {1↔5, 2↔6, 3↔8, 4↔7},
 {1↔5, 2↔7, 3↔4, 6↔8}, {1↔5, 2↔7, 3↔6, 4↔8}, {1↔5, 2↔7, 3↔8, 4↔6},
 {1↔5, 2↔8, 3↔4, 6↔7}, {1↔5, 2↔8, 3↔6, 4↔7}, {1↔5, 2↔8, 3↔7, 4↔6},
 {1↔6, 2↔3, 4↔5, 7↔8}, {1↔6, 2↔3, 4↔7, 5↔8}, {1↔6, 2↔3, 4↔8, 5↔7},
 {1↔6, 2↔4, 3↔5, 7↔8}, {1↔6, 2↔4, 3↔7, 5↔8}, {1↔6, 2↔4, 3↔8, 5↔7},
 {1↔6, 2↔5, 3↔4, 7↔8}, {1↔6, 2↔5, 3↔7, 4↔8}, {1↔6, 2↔5, 3↔8, 4↔7},
 {1↔6, 2↔7, 3↔4, 5↔8}, {1↔6, 2↔7, 3↔5, 4↔8}, {1↔6, 2↔7, 3↔8, 4↔5},
 {1↔6, 2↔8, 3↔4, 5↔7}, {1↔6, 2↔8, 3↔5, 4↔7}, {1↔6, 2↔8, 3↔7, 4↔5},
 {1↔7, 2↔3, 4↔5, 6↔8}, {1↔7, 2↔3, 4↔6, 5↔8}, {1↔7, 2↔3, 4↔8, 5↔6},
 {1↔7, 2↔4, 3↔5, 6↔8}, {1↔7, 2↔4, 3↔6, 5↔8}, {1↔7, 2↔4, 3↔8, 5↔6},
 {1↔7, 2↔5, 3↔4, 6↔8}, {1↔7, 2↔5, 3↔6, 4↔8}, {1↔7, 2↔5, 3↔8, 4↔6},
 {1↔7, 2↔6, 3↔4, 5↔8}, {1↔7, 2↔6, 3↔5, 4↔8}, {1↔7, 2↔6, 3↔8, 4↔5},
 {1↔7, 2↔8, 3↔4, 5↔6}, {1↔7, 2↔8, 3↔5, 4↔6}, {1↔7, 2↔8, 3↔6, 4↔5},
 {1↔8, 2↔3, 4↔5, 6↔7}, {1↔8, 2↔3, 4↔6, 5↔7}, {1↔8, 2↔3, 4↔7, 5↔6},
 {1↔8, 2↔4, 3↔5, 6↔7}, {1↔8, 2↔4, 3↔6, 5↔7}, {1↔8, 2↔4, 3↔7, 5↔6},
 {1↔8, 2↔5, 3↔4, 6↔7}, {1↔8, 2↔5, 3↔6, 4↔7}, {1↔8, 2↔5, 3↔7, 4↔6},
 {1↔8, 2↔6, 3↔4, 5↔7}, {1↔8, 2↔6, 3↔5, 4↔7}, {1↔8, 2↔6, 3↔7, 4↔5},
 {1↔8, 2↔7, 3↔4, 5↔6}, {1↔8, 2↔7, 3↔5, 4↔6}, {1↔8, 2↔7, 3↔6, 4↔5}}
```

In[●]:=

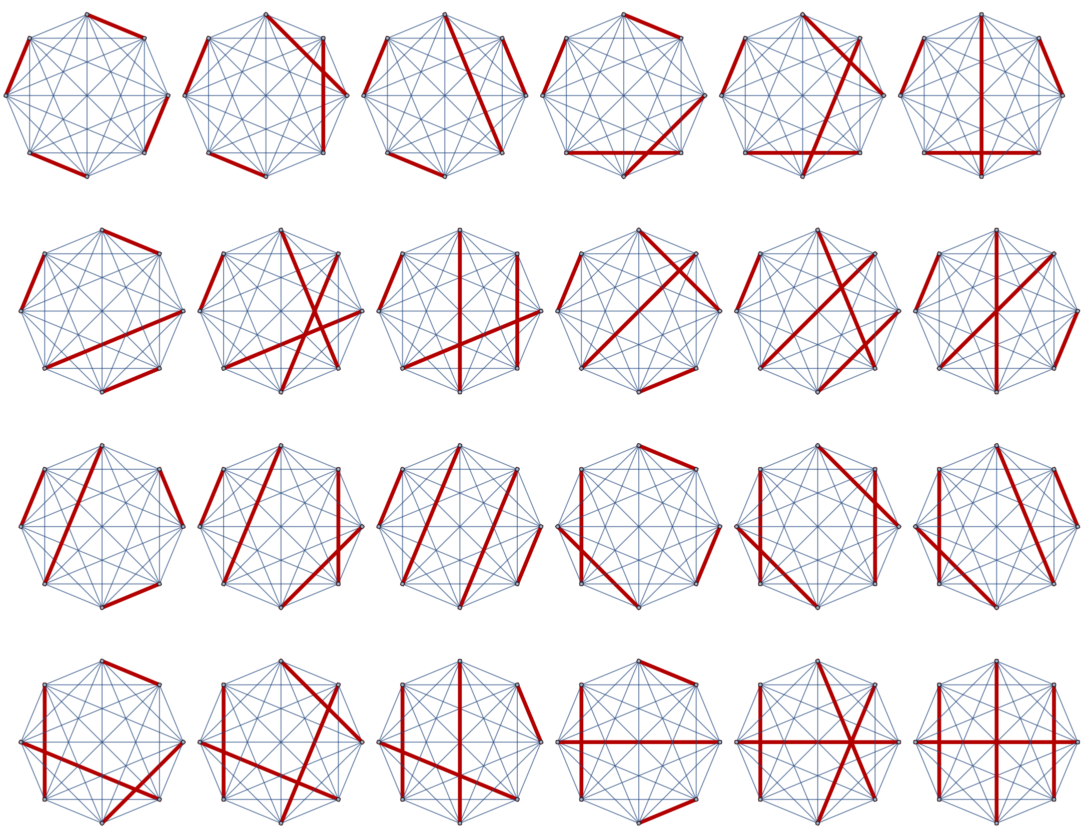
In[ ]:=
```
Grid[Table[HighlightGraph[k6, h, GraphHighlightStyle → "Thick"], {h, esl6}]]
```

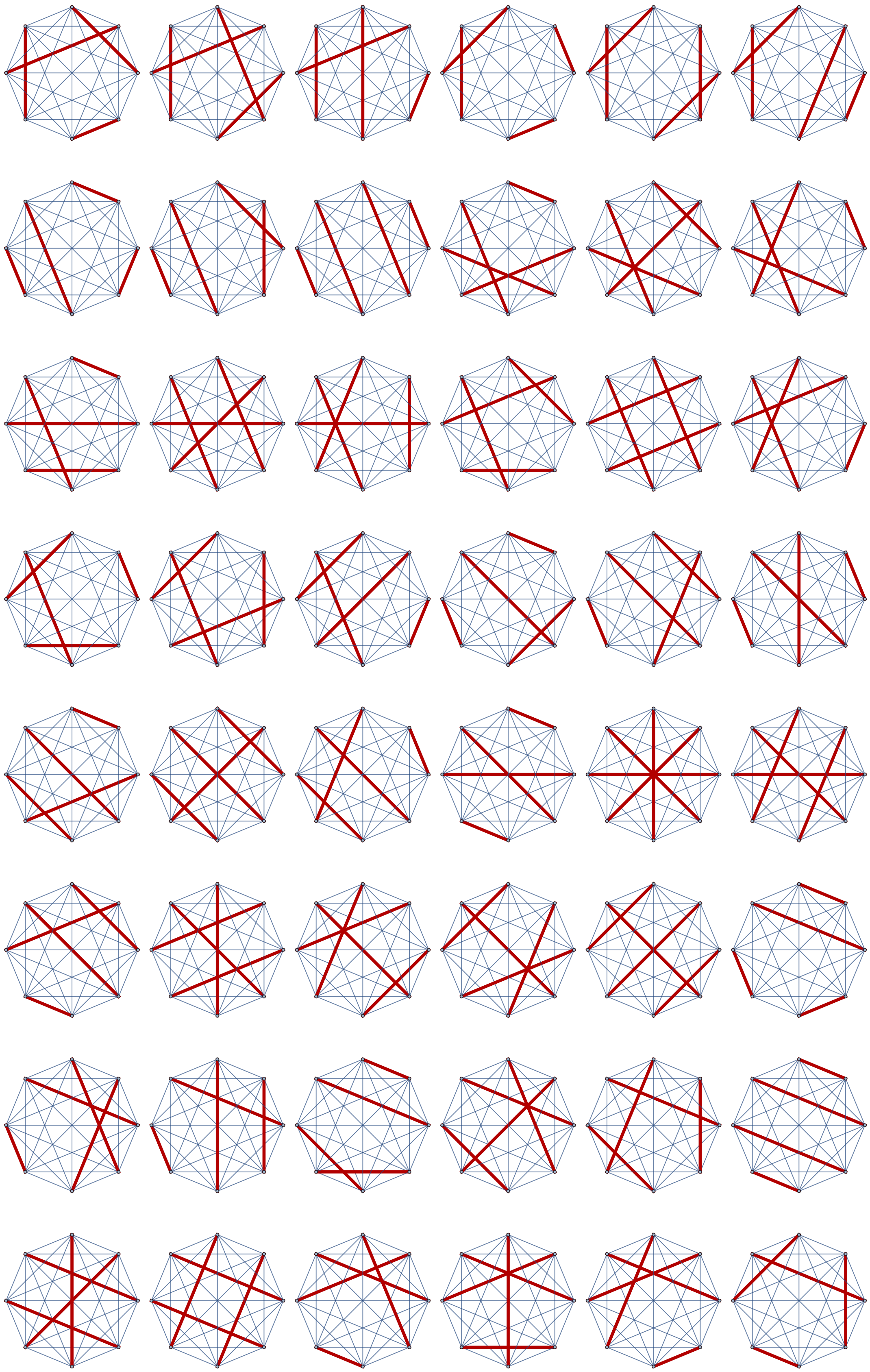
Out[ ]= Grid[{

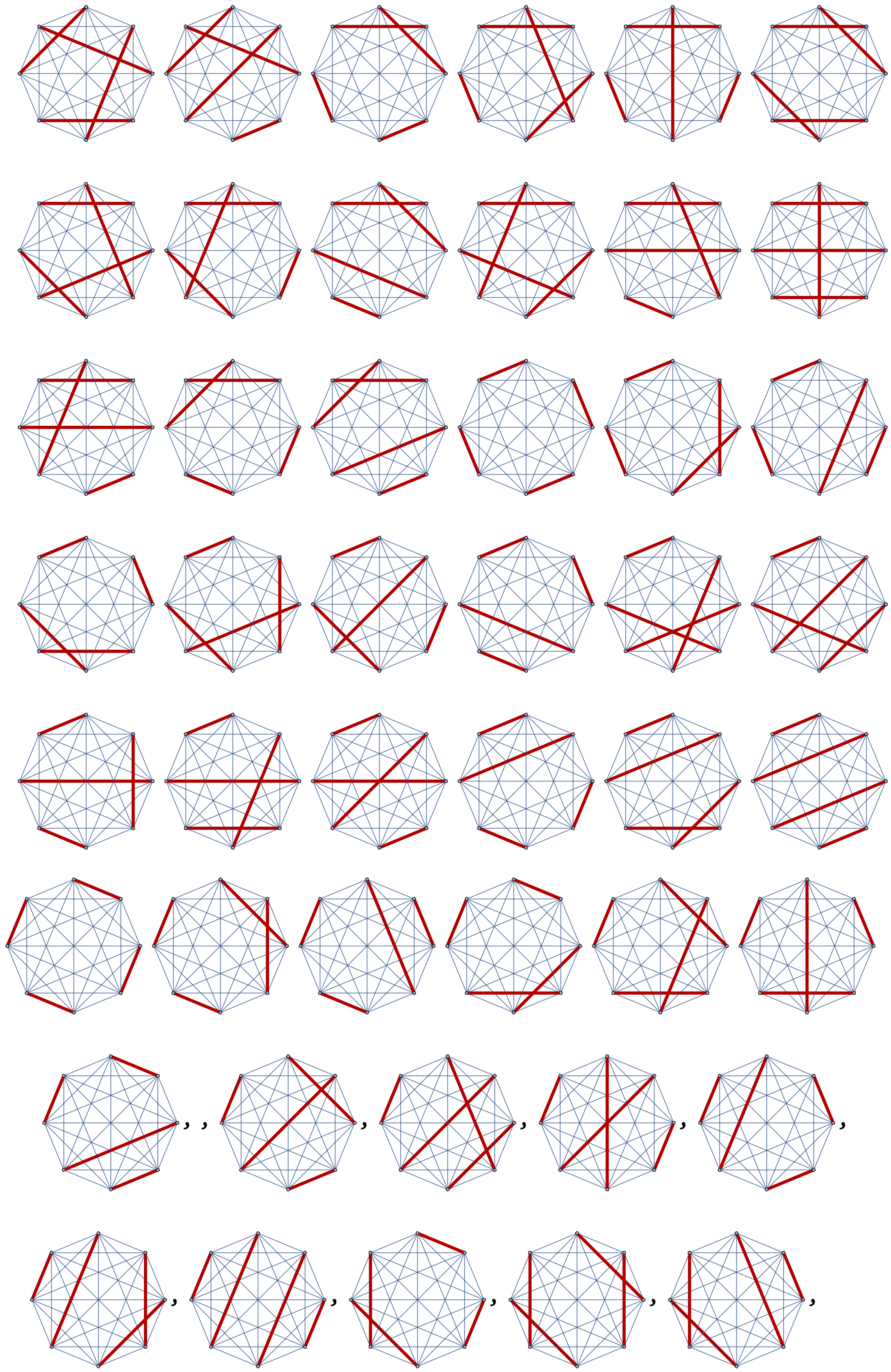


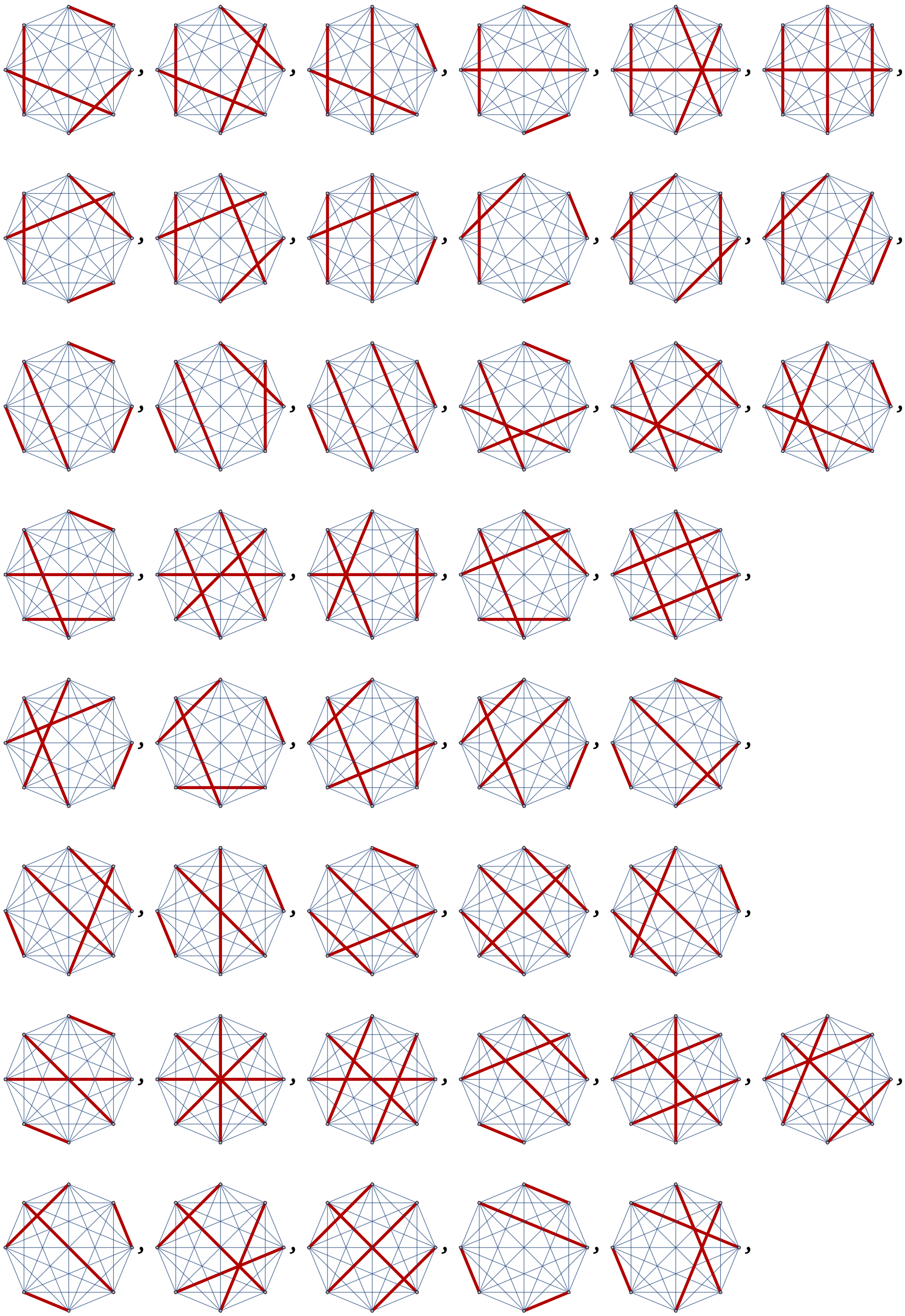
}]

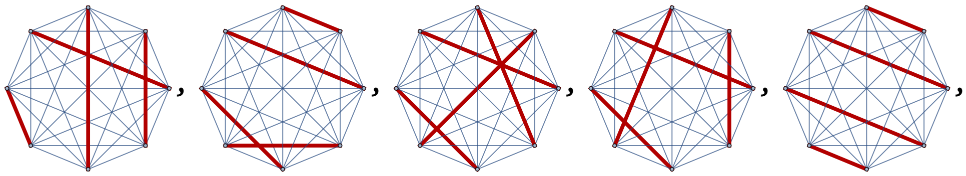
In[ ]:=
```
Grid[Table[HighlightGraph[k8, h, GraphHighlightStyle → "Thick"], {h, esl8}]]
```

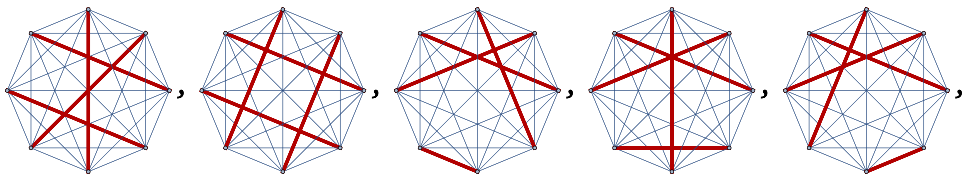


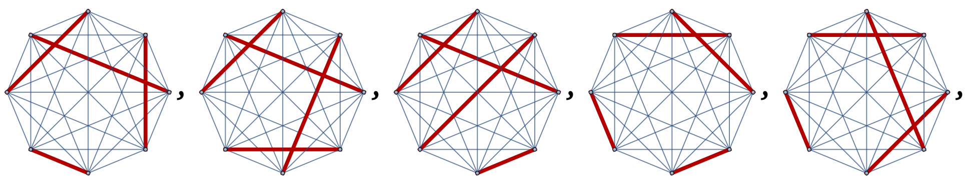


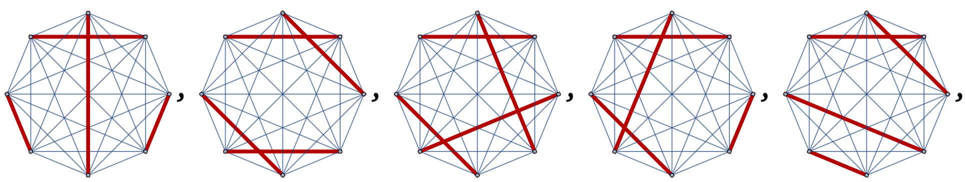


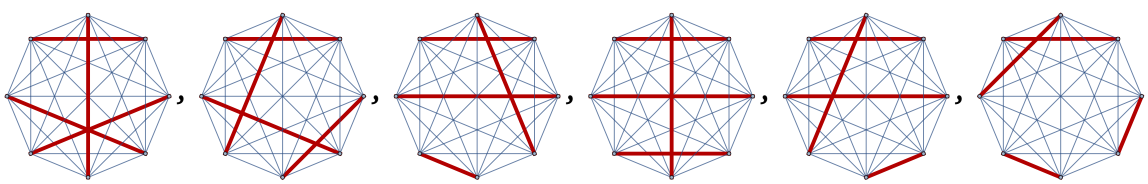


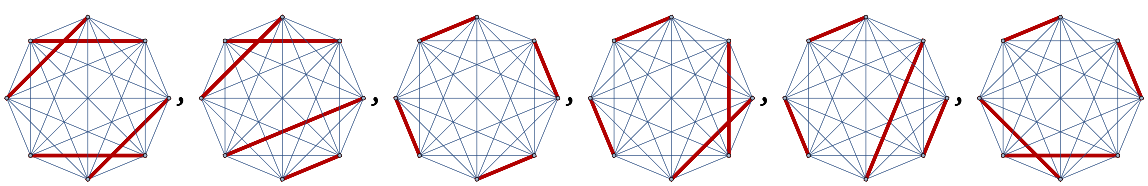


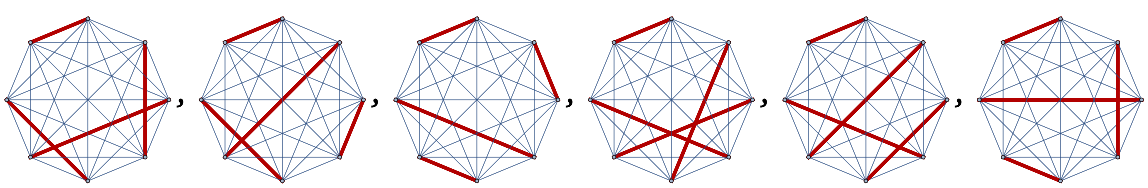


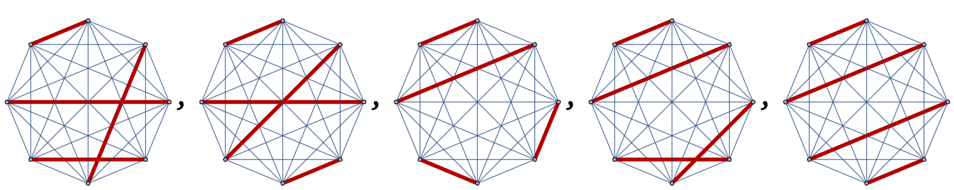












}]

Stirling Permutations

In[1]:=
```
ClearAll[strlngPermutations]
strlngPermutations[1] = {{1, 1}};
strlngPermutations[k_] := Join @@
  (Function[x, Flatten[Insert[x, {k, k}, #]] & /@ Range[2 k - 1]] /@ strlngPermutations[k - 1])
```

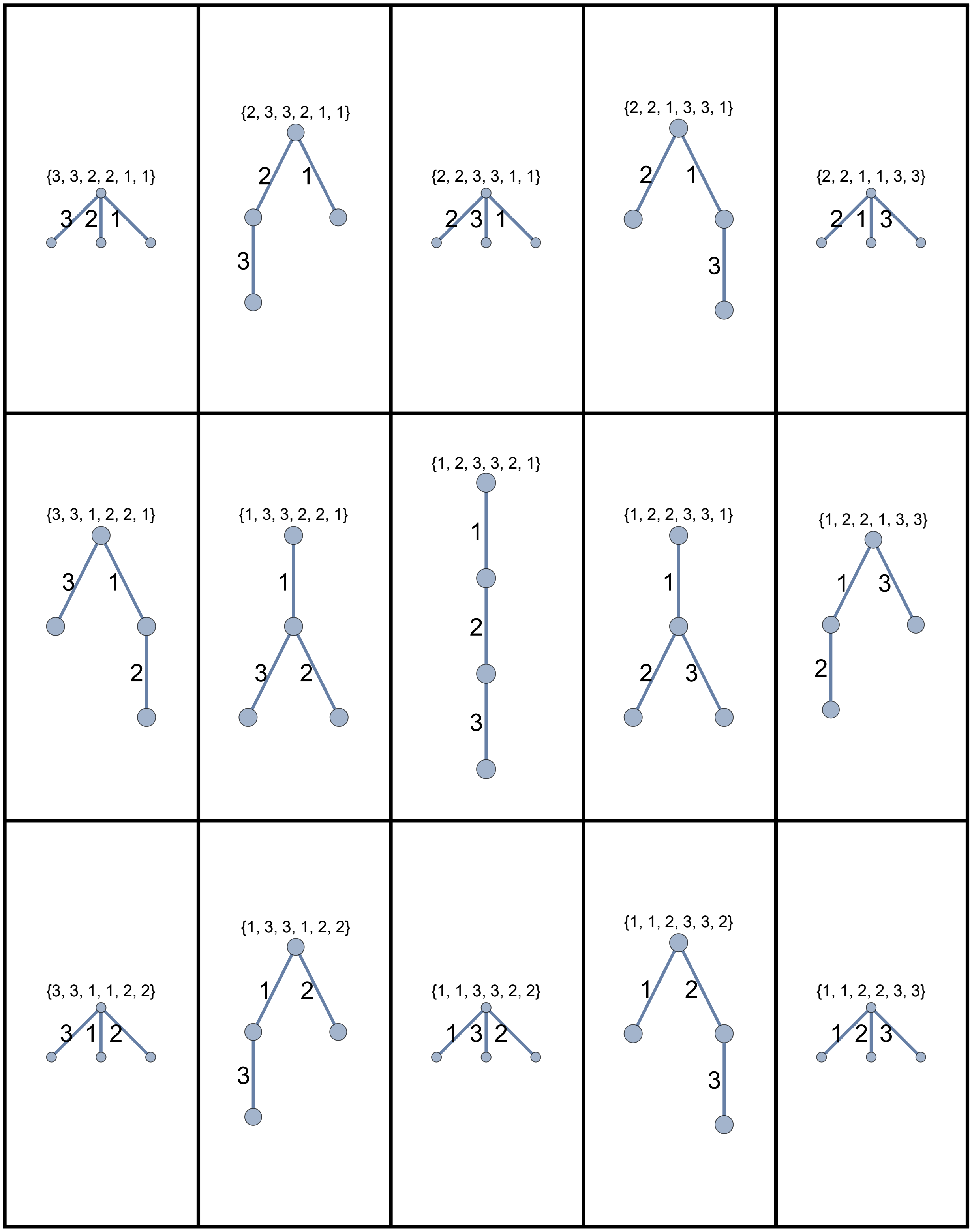
```
In[]:= Multicolumn[Sort@strlngPermutations@4, 5, Appearance → "Horizontal"]

Out[]=
{1, 1, 2, 2,     {1, 1, 2, 2,     {1, 1, 2, 2,     {1, 1, 2, 3,     {1, 1, 2, 3,
 3, 3, 4, 4}      3, 4, 4, 3}      4, 4, 3, 3}      3, 2, 4, 4}      3, 4, 4, 2}
{1, 1, 2, 3,     {1, 1, 2, 4,     {1, 1, 2, 4,     {1, 1, 3, 3,     {1, 1, 3, 3,
 4, 4, 3, 2}      4, 2, 3, 3}      4, 3, 3, 2}      2, 2, 4, 4}      2, 4, 4, 2}
{1, 1, 3, 3,     {1, 1, 3, 4,     {1, 1, 4, 4,     {1, 1, 4, 4,     {1, 1, 4, 4,
 4, 4, 2, 2}      4, 3, 2, 2}      2, 2, 3, 3}      2, 3, 3, 2}      3, 3, 2, 2}
{1, 2, 2, 1,     {1, 2, 2, 1,     {1, 2, 2, 1,     {1, 2, 2, 3,     {1, 2, 2, 3,
 3, 3, 4, 4}      3, 4, 4, 3}      4, 4, 3, 3}      3, 1, 4, 4}      3, 4, 4, 1}
{1, 2, 2, 3,     {1, 2, 2, 4,     {1, 2, 2, 4,     {1, 2, 3, 3,     {1, 2, 3, 3,
 4, 4, 3, 1}      4, 1, 3, 3}      4, 3, 3, 1}      2, 1, 4, 4}      2, 4, 4, 1}
{1, 2, 3, 3,     {1, 2, 3, 4,     {1, 2, 4, 4,     {1, 2, 4, 4,     {1, 2, 4, 4,
 4, 4, 2, 1}      4, 3, 2, 1}      2, 1, 3, 3}      2, 3, 3, 1}      3, 3, 2, 1}
{1, 3, 3, 1,     {1, 3, 3, 1,     {1, 3, 3, 1,     {1, 3, 3, 2,     {1, 3, 3, 2,
 2, 2, 4, 4}      2, 4, 4, 2}      4, 4, 2, 2}      2, 1, 4, 4}      2, 4, 4, 1}
{1, 3, 3, 2,     {1, 3, 3, 4,     {1, 3, 3, 4,     {1, 3, 4, 4,     {1, 3, 4, 4,
 4, 4, 2, 1}      4, 1, 2, 2}      4, 2, 2, 1}      3, 1, 2, 2}      3, 2, 2, 1}
{1, 4, 4, 1,     {1, 4, 4, 1,     {1, 4, 4, 1,     {1, 4, 4, 2,     {1, 4, 4, 2,
 2, 2, 3, 3}      2, 3, 3, 2}      3, 3, 2, 2}      2, 1, 3, 3}      2, 3, 3, 1}
{1, 4, 4, 2,     {1, 4, 4, 3,     {1, 4, 4, 3,     {2, 2, 1, 1,     {2, 2, 1, 1,
 3, 3, 2, 1}      3, 1, 2, 2}      3, 2, 2, 1}      3, 3, 4, 4}      3, 4, 4, 3}
{2, 2, 1, 1,     {2, 2, 1, 3,     {2, 2, 1, 3,     {2, 2, 1, 3,     {2, 2, 1, 4,
 4, 4, 3, 3}      3, 1, 4, 4}      3, 4, 4, 1}      4, 4, 3, 1}      4, 1, 3, 3}
{2, 2, 1, 4,     {2, 2, 3, 3,     {2, 2, 3, 3,     {2, 2, 3, 3,     {2, 2, 3, 4,
 4, 3, 3, 1}      1, 1, 4, 4}      1, 4, 4, 1}      4, 4, 1, 1}      4, 3, 1, 1}
{2, 2, 4, 4,     {2, 2, 4, 4,     {2, 2, 4, 4,     {2, 3, 3, 2,     {2, 3, 3, 2,
 1, 1, 3, 3}      1, 3, 3, 1}      3, 3, 1, 1}      1, 1, 4, 4}      1, 4, 4, 1}
{2, 3, 3, 2,     {2, 3, 3, 4,     {2, 3, 4, 4,     {2, 4, 4, 2,     {2, 4, 4, 2,
 4, 4, 1, 1}      4, 2, 1, 1}      3, 2, 1, 1}      1, 1, 3, 3}      1, 3, 3, 1}
{2, 4, 4, 2,     {2, 4, 4, 3,     {3, 3, 1, 1,     {3, 3, 1, 1,     {3, 3, 1, 1,
 3, 3, 1, 1}      3, 2, 1, 1}      2, 2, 4, 4}      2, 4, 4, 2}      4, 4, 2, 2}
{3, 3, 1, 2,     {3, 3, 1, 2,     {3, 3, 1, 2,     {3, 3, 1, 4,     {3, 3, 1, 4,
 2, 1, 4, 4}      2, 4, 4, 1}      4, 4, 2, 1}      4, 1, 2, 2}      4, 2, 2, 1}
{3, 3, 2, 2,     {3, 3, 2, 2,     {3, 3, 2, 2,     {3, 3, 2, 4,     {3, 3, 4, 4,
 1, 1, 4, 4}      1, 4, 4, 1}      4, 4, 1, 1}      4, 2, 1, 1}      1, 1, 2, 2}
{3, 3, 4, 4,     {3, 3, 4, 4,     {3, 4, 4, 3,     {3, 4, 4, 3,     {3, 4, 4, 3,
 1, 2, 2, 1}      2, 2, 1, 1}      1, 1, 2, 2}      1, 2, 2, 1}      2, 2, 1, 1}
{4, 4, 1, 1,     {4, 4, 1, 1,     {4, 4, 1, 1,     {4, 4, 1, 2,     {4, 4, 1, 2,
 2, 2, 3, 3}      2, 3, 3, 2}      3, 3, 2, 2}      2, 1, 3, 3}      2, 3, 3, 1}
{4, 4, 1, 2,     {4, 4, 1, 3,     {4, 4, 1, 3,     {4, 4, 2, 2,     {4, 4, 2, 2,
 3, 3, 2, 1}      3, 1, 2, 2}      3, 2, 2, 1}      1, 1, 3, 3}      1, 3, 3, 1}
{4, 4, 2, 2,     {4, 4, 2, 3,     {4, 4, 3, 3,     {4, 4, 3, 3,     {4, 4, 3, 3,
 3, 3, 1, 1}      3, 2, 1, 1}      1, 1, 2, 2}      1, 2, 2, 1}      2, 2, 1, 1}
```

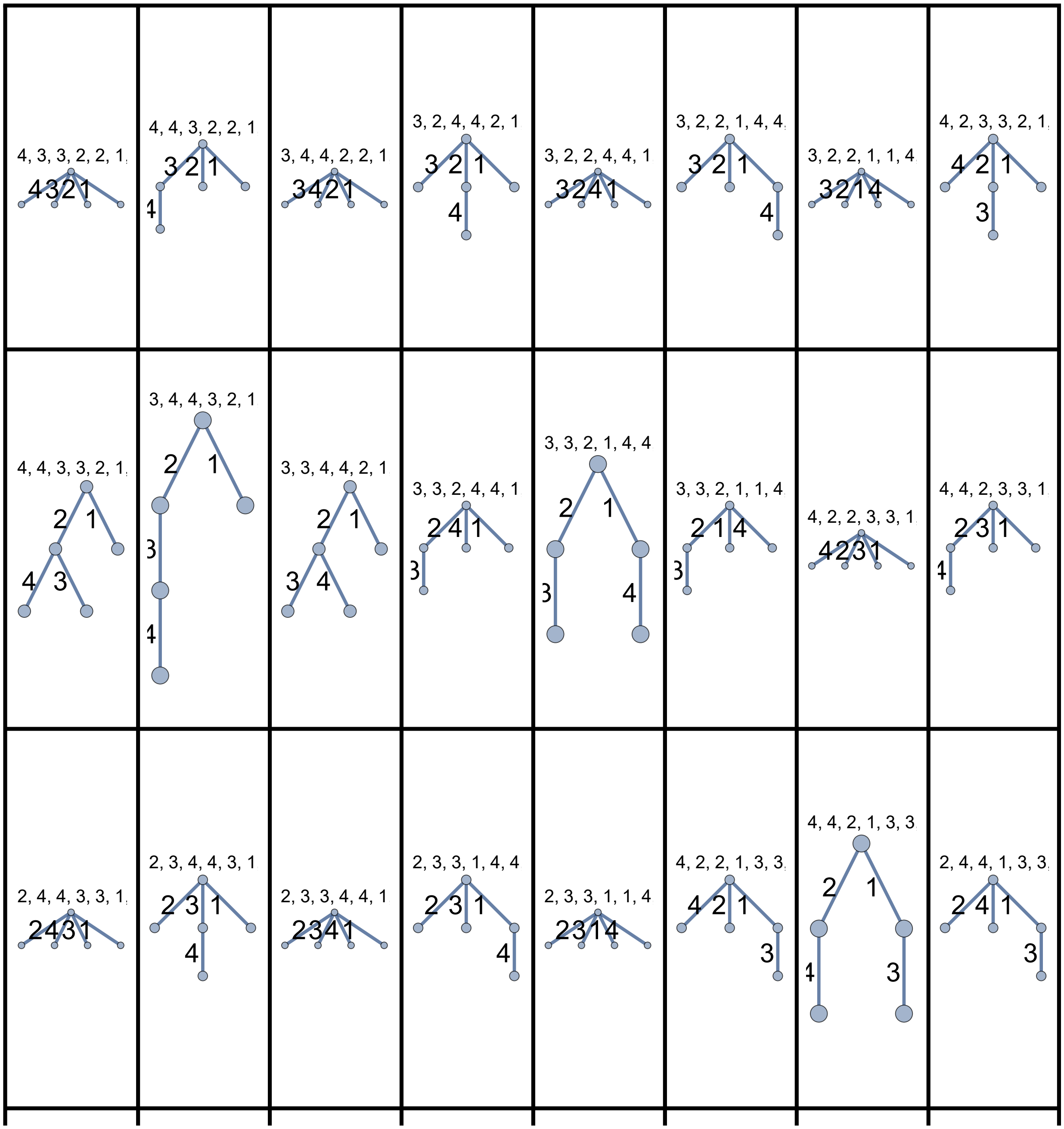
In[4]:=
```
ClearAll[strlngPermGraph]
strlngPermGraph[sp_, opts : OptionsPattern[]] :=
 Module[{vl = DeleteDuplicates@sp, pos = PositionIndex@sp,
   eL = EdgeList@*TransitiveReductionGraph@*GraphUnion},
  Graph[Prepend[vl, 0], eL[Graph@Thread[0 → vl],
     SimpleGraph@RelationGraph[And @@ Between[pos@#] /@ pos[#2] &, vl]],
   GraphLayout → {"LayeredEmbedding", "RootVertex" → 0},
   EdgeLabels → {e_ :→ Placed[Last@e, {Left, "Middle"}]}, opts]]
```

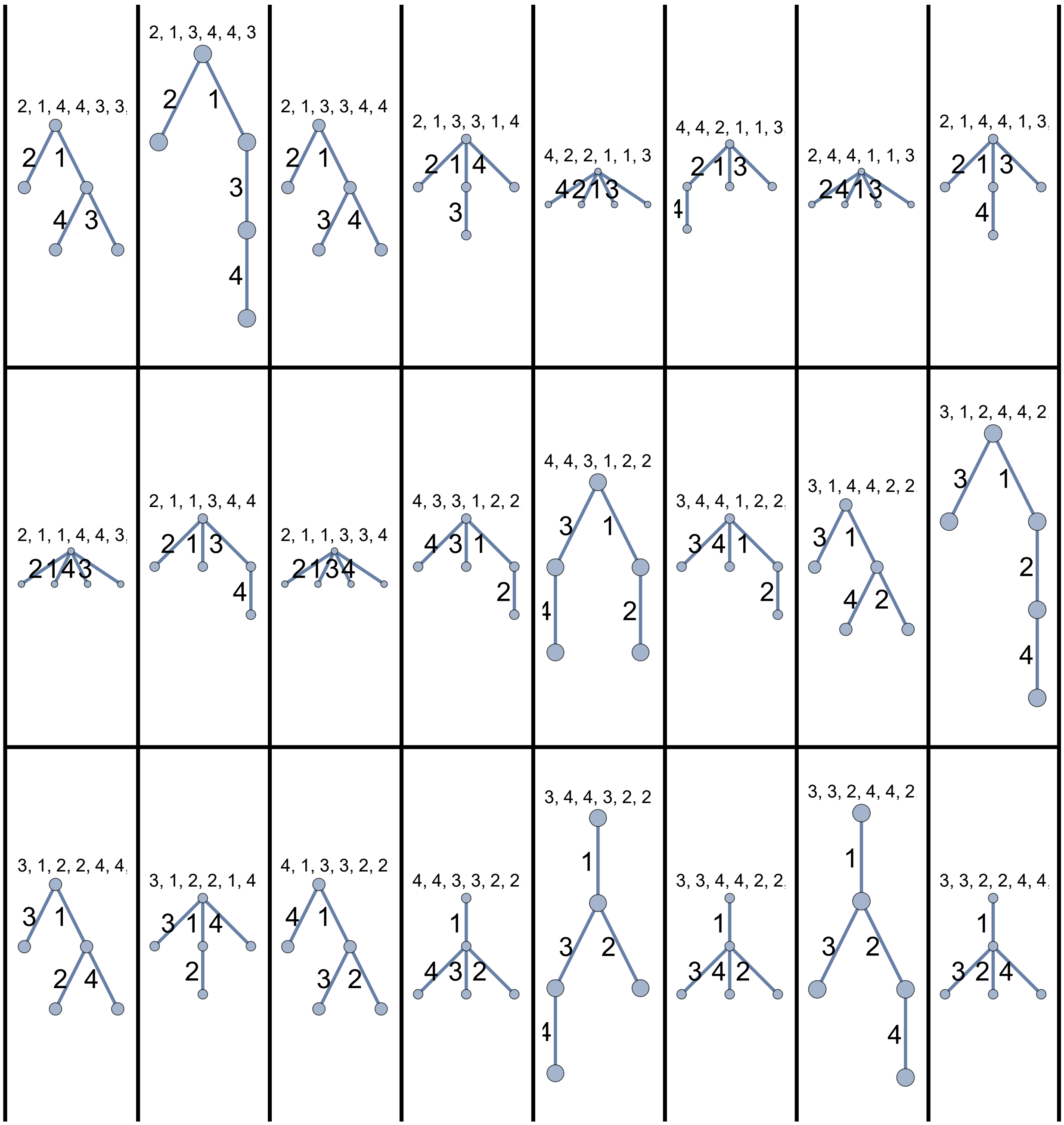
In[ ]:=
```
Grid[Partition[strlngPermGraph[#, PlotLabel → #, EdgeShapeFunction → "Line",
      EdgeStyle → Thick, EdgeLabelStyle → 16, VertexSize → Medium] & /@
    strlngPermutations[3], 5], Dividers → All, Spacings → {4, 4}]
```

Out[ ]=

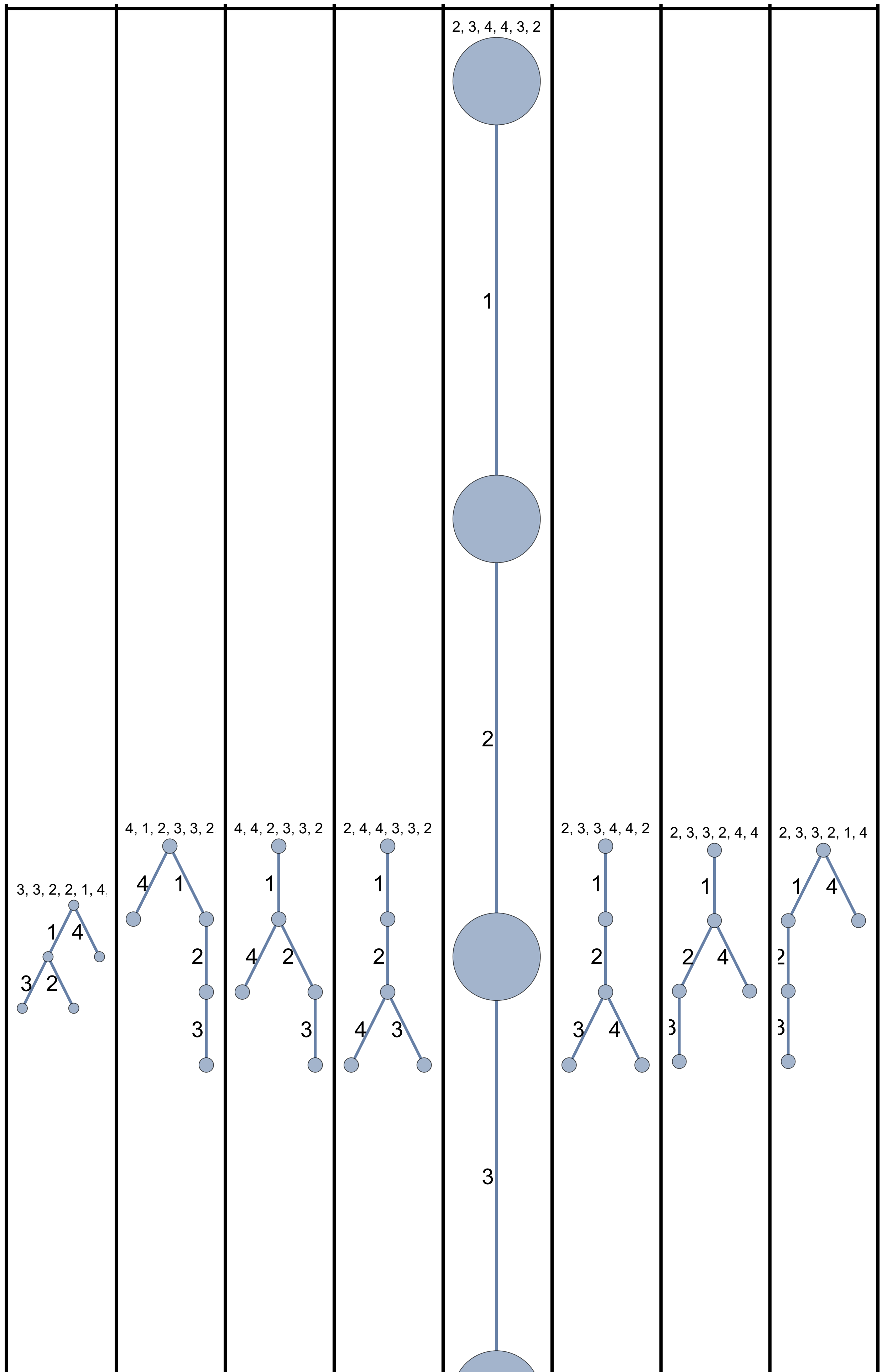


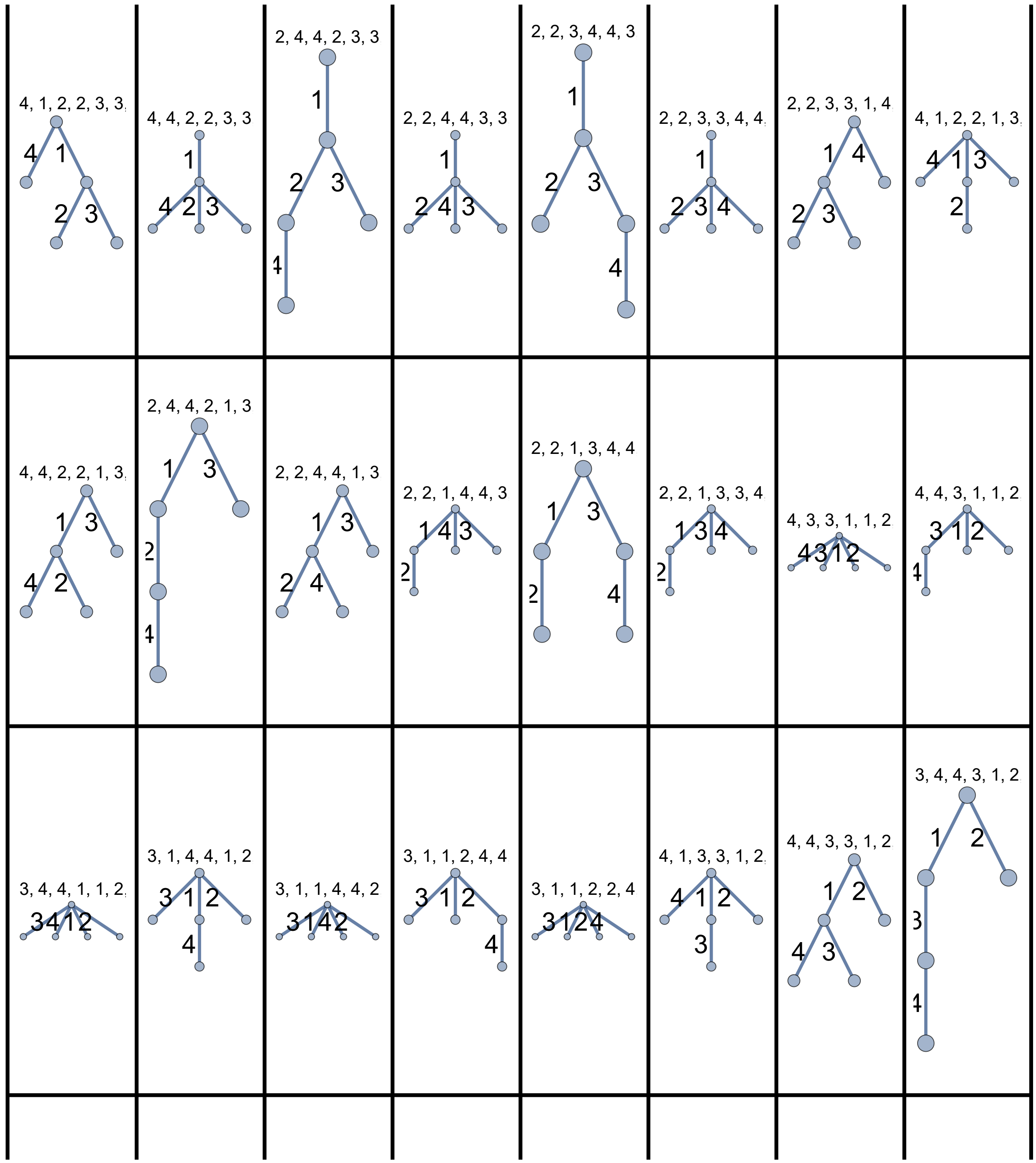
In[9]:=
```
Grid[Partition[strlngPermGraph[#, PlotLabel → #, EdgeShapeFunction → "Line",
      EdgeStyle → Thick, EdgeLabelStyle → 16, VertexSize → Medium] & /@
    strlngPermutations[4], 8], Dividers → All, Spacings → {1, 1}]
```

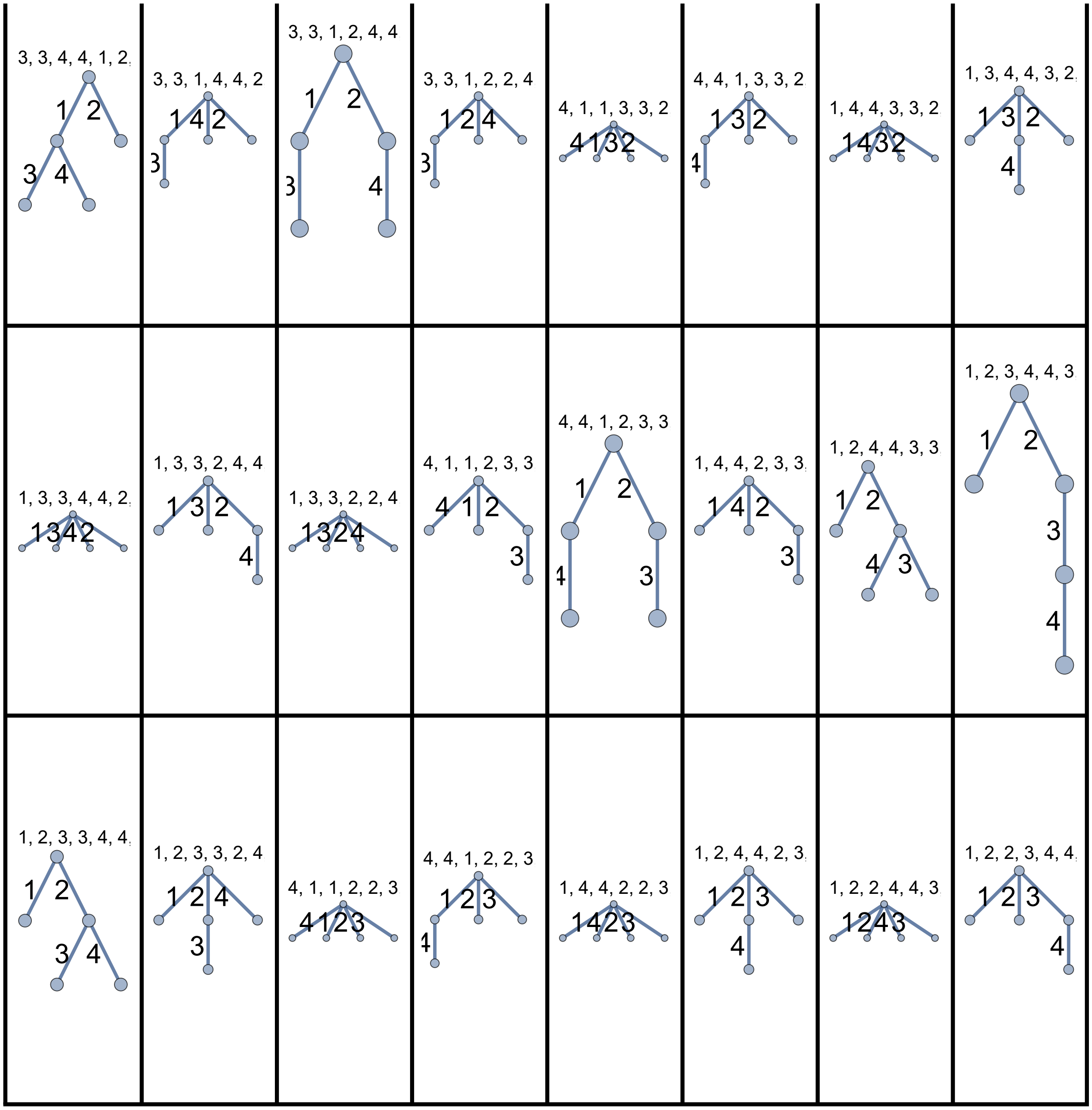



2, 1, 4, 4, 3, 3,
2, 1, 3, 4, 4, 3
2, 1, 3, 3, 4, 4
2, 1, 3, 3, 1, 4
4, 2, 2, 1, 1, 3
4, 4, 2, 1, 1, 3
2, 4, 4, 1, 1, 3
2, 1, 4, 4, 1, 3,
2, 1, 1, 4, 4, 3,
2, 1, 1, 3, 4, 4
2, 1, 1, 3, 3, 4
4, 3, 3, 1, 2, 2
4, 4, 3, 1, 2, 2
3, 4, 4, 1, 2, 2,
3, 1, 4, 4, 2, 2
3, 1, 2, 4, 4, 2,
3, 1, 2, 2, 4, 4,
3, 1, 2, 2, 1, 4
4, 1, 3, 3, 2, 2
4, 4, 3, 3, 2, 2
3, 4, 4, 3, 2, 2
3, 3, 4, 4, 2, 2,
3, 3, 2, 4, 4, 2
3, 3, 2, 2, 4, 4,

Out[9]=




4, 1, 2, 2, 3, 3,
4, 4, 2, 2, 3, 3
2, 4, 4, 2, 3, 3
2, 2, 4, 4, 3, 3
2, 2, 3, 4, 4, 3
2, 2, 3, 3, 4, 4
2, 2, 3, 3, 1, 4
4, 1, 2, 2, 1, 3,
4, 4, 2, 2, 1, 3,
2, 4, 4, 2, 1, 3
2, 2, 4, 4, 1, 3
2, 2, 1, 4, 4, 3
2, 2, 1, 3, 4, 4
2, 2, 1, 3, 3, 4
4, 3, 3, 1, 1, 2
4, 4, 3, 1, 1, 2,
3, 4, 4, 1, 1, 2,
3, 1, 4, 4, 1, 2
3, 1, 1, 4, 4, 2
3, 1, 1, 2, 4, 4
3, 1, 1, 2, 2, 4
4, 1, 3, 3, 1, 2,
4, 4, 3, 3, 1, 2
3, 4, 4, 3, 1, 2,



Limits for Ratio test in m(2)

In[ ]:=
```
case1 = (2^(1 + 2 n) n! (1 + n)!) / (2 (1 + n))!
case2 = (2^(-1 - 2 n) (2 n)!) / (n! (1 + n)!)
```

Out[ ]=
$$\frac{2^{1+2n}\, n!\, (1+n)!}{(2\,(1+n))!}$$

Out[ ]=
$$\frac{2^{-1-2n}\, (2n)!}{n!\, (1+n)!}$$

```
In[•]:= limitm2case1 = Limit[case1, n → Infinity]
        limitm2case2 = Limit[case2, n → Infinity]
Out[•]= 0
Out[•]= 0
```

Computing m(1) to m(10) from n=0 to n=2000

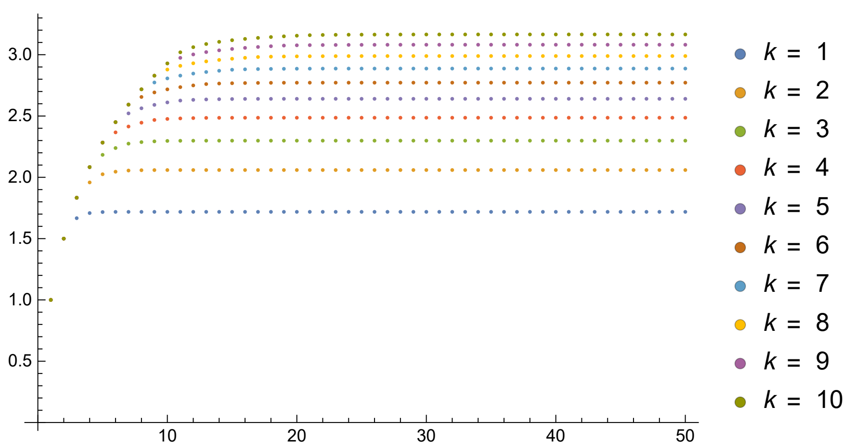
```
In[•]:= Multifactorial[n_, k_] := Abs[Apply[Times, Range[-n, -1, k]]]
```

```
In[•]:= For[i = 1, i < 11, i++, Print[N[Sum[1 / Multifactorial[n, i], {n, 0, 150}], 20]]]
2.7182818284590452354
3.0594074053425761445
3.2989135380884190034
3.4859449774535577452
3.6402244677338097342
3.7719023962117584357
3.8869596537408434954
3.9892412126901365441
4.0813755201688985441
4.1652437655583845908
```

Plot of m (1) to m (1) superimposed on each other from n = 0 to n = 2000

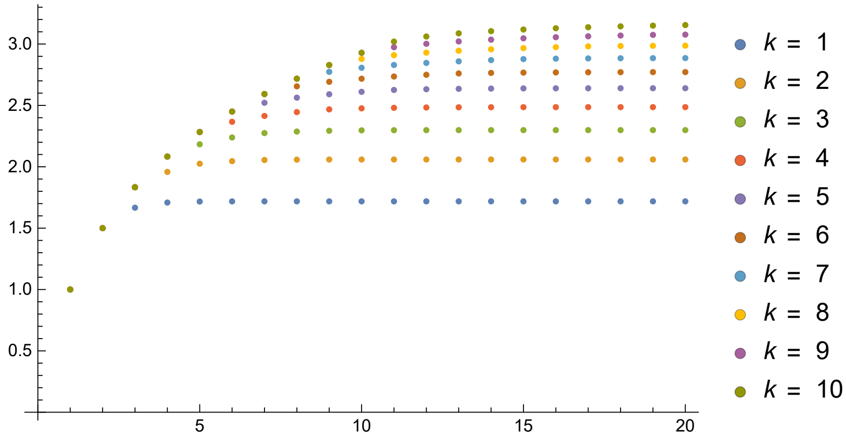
```
ListPlot[Table[Sum[1 / Multifactorial[n, j], {n, 1, i}], {j, 1, 10}, {i, 1, 20}],
 PlotLegends → PointLegend[Automatic,
   PromptForm[k, #] & /@ Range[10], LegendMarkers → {Graphics[Disk[]], 6}]]
```

Out[•]=



```
ListPlot[Table[Sum[1 / Multifactorial[n, j], {n, 1, i}], {j, 1, 10}, {i, 1, 50}],
 PlotLegends → PointLegend[Automatic,
   PromptForm[k, #] & /@ Range[10], LegendMarkers → {Graphics[Disk[]], 6}]]
```

Out[ ]=



Computation of RMFCs using the closed form formula

In[ ]:=

$$\text{ClosedFormRMFC}[n\_] := 1 + \frac{1}{n}\,\text{Exp}[1/n]\ \text{Sum}\left[n^{k/n}\,\text{Gamma}\left[\frac{k}{n}, 0, \frac{1}{n}\right], \{k, n\}\right]$$

In[ ]:=

```
For[i = 1, i < 11, i++, Print[N[ClosedFormRMFC[i], 20]]]
```

```
2.7182818284590452354
3.0594074053425761445
3.2989135380884190034
3.4859449774535577452
3.6402244677338097342
3.7719023962117584357
3.8869596537408434954
3.9892412126901365441
4.0813755201688985441
4.1652437655583845908
```

In[ ]:=

```
N[ClosedFormRMFC[100], 100]
```

Out[ ]=

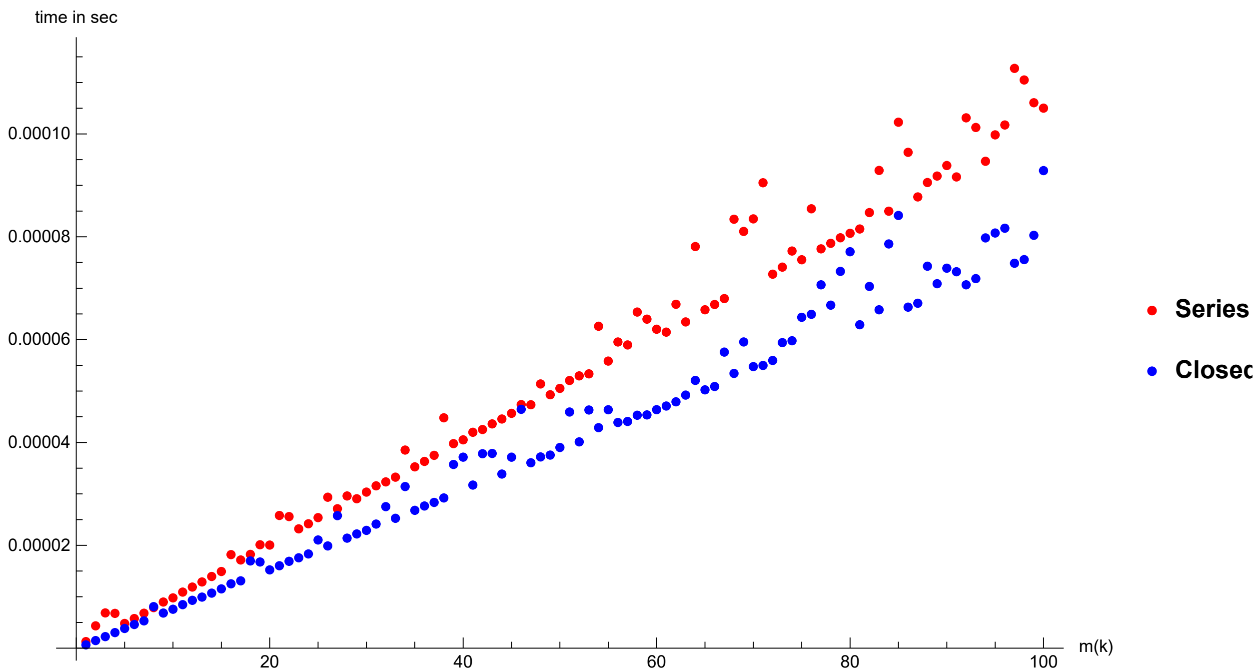
```
6.23255596900487817559483331447484719147772171073261303698458651259212659259992188311253
 0488409940747
```

Analysing efficiency of the two RMFC calculation methods

```
In[•]:= test1[xx_] :=
         (For[i = 1, i < xx, i++, Print[N[Sum[1 / Multifactorial[n, i], {n, 0, 250}], 50]] //
             Inactive] // RepeatedTiming) [[1]]
        test2[xx_] := (For[i = 1, i < xx, i++, Print[N[ClosedFormRMFC[i], 50]] // Inactive] //
             RepeatedTiming) [[1]]
```

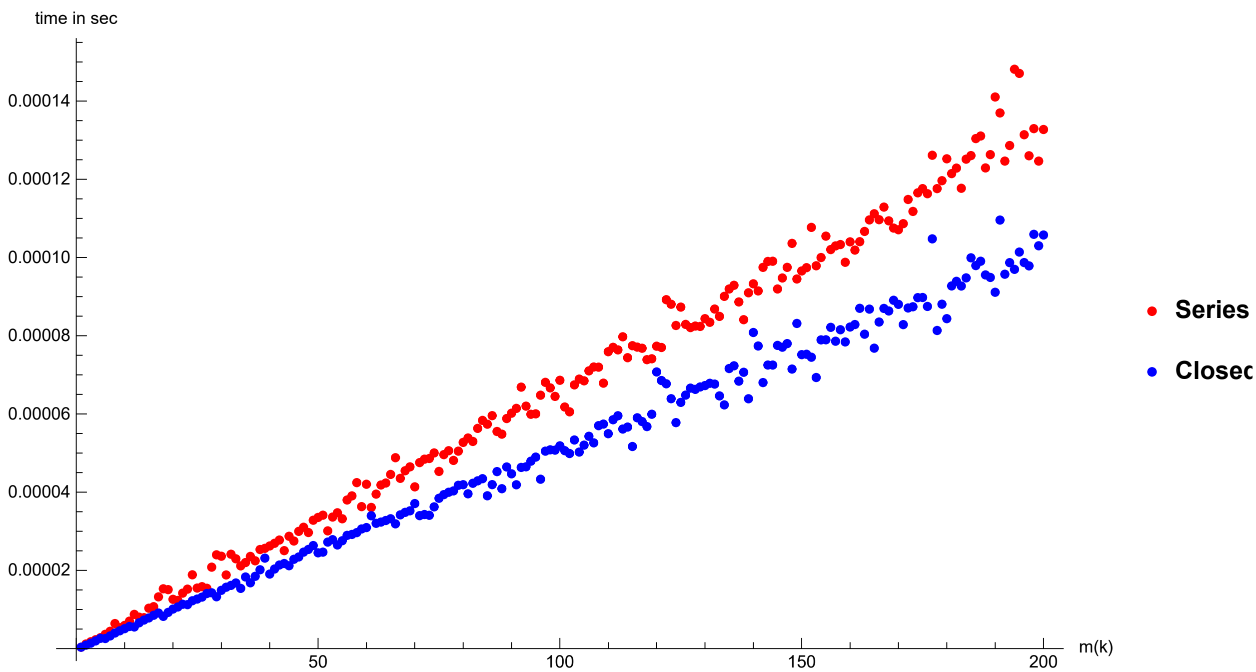
```
In[•]:= list1 =
          ListPlot[Table[{xx, test1[xx]}, {xx, 1, 100}], PlotRange → All, PlotLegends → {Automatic},
           AxesLabel → {"m(k)", "time in sec"}, PlotStyle → {Red, Thick}];
        list2 = ListPlot[Table[{xx, test2[xx]}, {xx, 1, 100}], PlotRange → All, PlotLegends →
             {Automatic}, AxesLabel → {"m(k)", "time in sec"}, PlotStyle → {Blue, Thick}];
```

```
In[•]:= Show[list1, list2, ImageSize → Large]
```




```
In[•]:= list3 =
          ListPlot[Table[{xx, test1[xx]}, {xx, 1, 200}], PlotRange → All, PlotLegends → {Automatic},
           AxesLabel → {"m(k)", "time in sec"}, PlotStyle → {Red, Thick}];
        list4 = ListPlot[Table[{xx, test2[xx]}, {xx, 1, 200}], PlotRange → All, PlotLegends →
             {Automatic}, AxesLabel → {"m(k)", "time in sec"}, PlotStyle → {Blue, Thick}];
```

```
In[•]:= Show[list3, list4, ImageSize → Large]
```

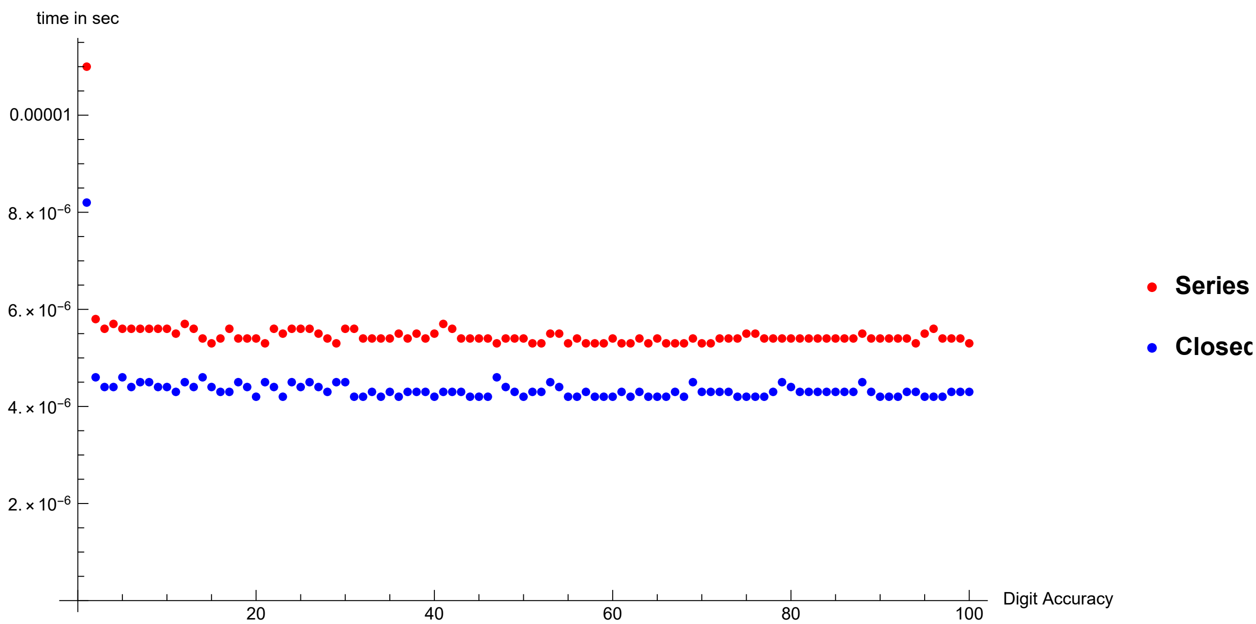



```
test3[xx_] :=
  (For[i = 1, i < 10, i++, Print[N[Sum[1 / Multifactorial[n, i], {n, 0, 500}], xx]] //
      Inactive] // AbsoluteTiming) [[1]]
test4[xx_] := (For[i = 1, i < 10, i++, Print[N[ClosedFormRMFC[i], xx]] // Inactive] //
     AbsoluteTiming) [[1]]
```

```
In[]:= list5 =
    ListPlot[Table[{xx, test3[xx]}, {xx, 1, 100}], PlotRange → All, PlotLegends → {Automatic},
     AxesLabel → {"Digit Accuracy", "time in sec"}, PlotStyle → {Red, Thick}];
```

```
In[]:= list6 =
    ListPlot[Table[{xx, test4[xx]}, {xx, 1, 100}], PlotRange → All, PlotLegends → {Automatic},
     AxesLabel → {"Digit Accuracy", "time in sec"}, PlotStyle → {Blue, Thick}];
```

```
In[]:= Show[list5, list6, ImageSize → Large]
```



In[◦]:= list7 =

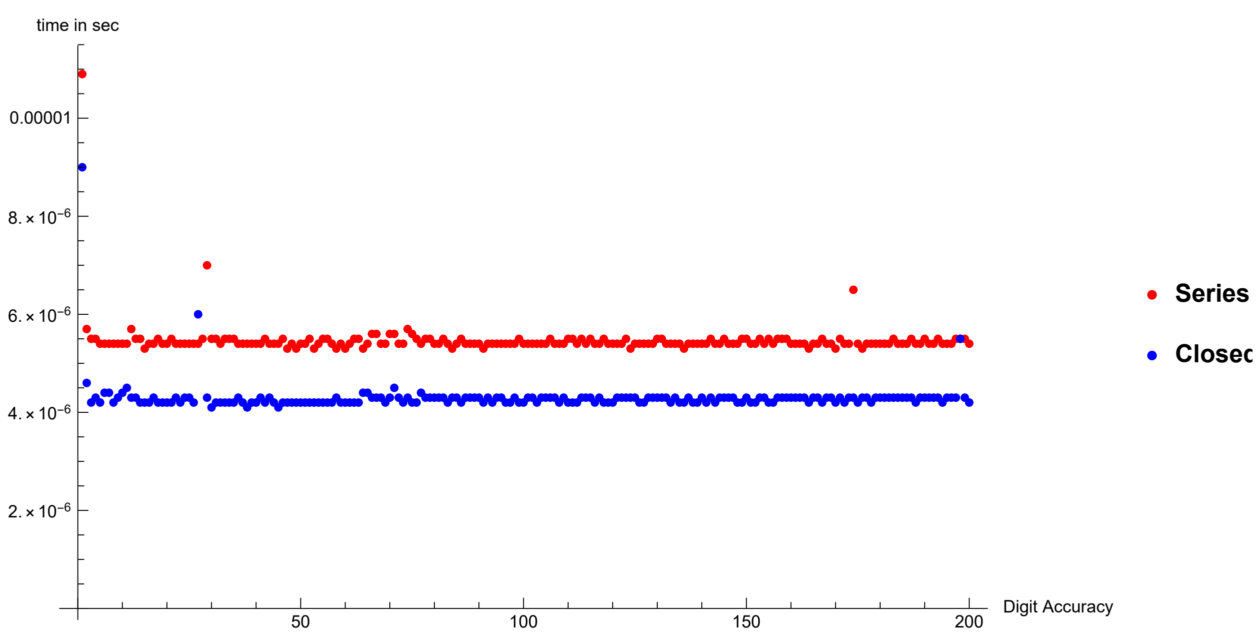
```
list7 =
  ListPlot[Table[{xx, test3[xx]}, {xx, 1, 200}], PlotRange → All, PlotLegends → {Automatic},
   AxesLabel → {"Digit Accuracy", "time in sec"}, PlotStyle → {Red, Thick}];
```

In[◦]:=

```
list8 =
  ListPlot[Table[{xx, test4[xx]}, {xx, 1, 200}], PlotRange → All, PlotLegends → {Automatic},
   AxesLabel → {"Digit Accuracy", "time in sec"}, PlotStyle → {Blue, Thick}];
```

In[◦]:=

```
Show[list7, list8, ImageSize → Large]
```

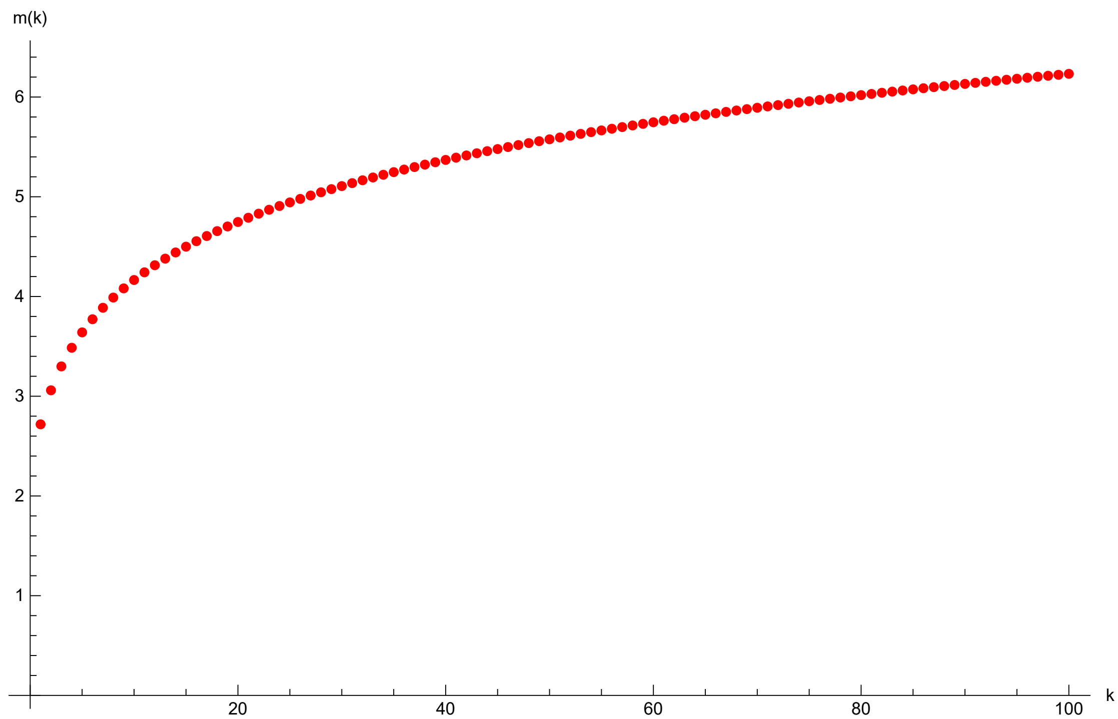


Asymptotics of Reciprocal Multifactorial Series

Simple Graph

In[ ]:=
```
ListPlot[Table[{x, ClosedFormRMFC[x]}, {x, 1, 100}],
 ImageSize → Large, AxesLabel → {"k", "m(k)"}, PlotStyle → {Red, Thick}]
```

Out[ ]=



Detailed asymptotics

In[ ]:=
```
Asymptotic[λ Integrate[((1 - t) (1 - E^(-(λ t)))) / (t (E^(-(λ Log[t])) - 1)), {t, 0, 1}],
 λ → 0, SeriesTermGoal → 1]
```

Out[ ]= $\lambda\, \mathrm{Log}[2]$

```
Asymptotic[λ Integrate[((1 - t) (1 - E^(-(λ t)))) / (t (E^(-(λ Log[t])) - 1)), {t, 0, 1}],
 λ → 0, SeriesTermGoal → 2]
```

Out[ ]= $-\frac{\lambda^2}{4} - \frac{1}{2}\lambda^2\, \mathrm{Log}[3] + \frac{1}{2}\lambda\, \mathrm{Log}[4] + \frac{1}{12}\lambda^2\, \mathrm{Log}[64]$

In[ ]:=
```
Asymptotic[HarmonicNumber[k], k → Infinity, SeriesTermGoal → 2]
```

Out[ ]= $\mathrm{EulerGamma} - \frac{1}{12\, k^2} + \frac{1}{2\, k} + \mathrm{Log}[k]$

In[•]:=
```
FullSimplify[1 + (1 + 1/(2 k^2) + 1/k) (EulerGamma + 1/(6 k^2) + 1/(2 k) + Log[3]/(2 k^2) - Log[4]/(2 k) - Log[64]/(12 k^2) + Log[k])]
```

Out[•]=
$$1+\frac{(1+2 k (1+k)) \left(1+\text{Log}\left[\frac{27}{8}\right]+k (3+6 \text{ EulerGamma } k-\text{Log}[64])+6 k^2 \text{Log}[k]\right)}{12 k^4}$$

In[•]:=
```
FullSimplify[1 + (1 + 1/k) * ((EulerGamma + Log[k] + 1/(2 k)) - (1/k Log[2]))]
```

Out[•]=
$$1+\frac{(1+k) (1+2 \text{ EulerGamma } k-\text{Log}[4]+2 k \text{ Log}[k])}{2 k^2}$$

In[•]:=
```
ClosedFormRMFC[n_] := 1 + 1/n Exp[1/n] Sum[n^(k/n) Gamma[k/n, 0, 1/n], {k, n}]
```

In[•]:=
```
RMFCApproximation[k_] :=
  1 + ((1 + 2 k (1 + k)) (1 + Log[27/8] + k (3 + 6 EulerGamma k - Log[64]) + 6 k^2 Log[k]))/(12 k^4)
RMFCApproximation1[k_] := 1 + ((1 + k) (1 + 2 EulerGamma k - 2 Log[2] + 2 k Log[k]))/(2 k^2)
```

In[•]:=
```
N[RMFCApproximation[15]]
N[RMFCApproximation1[15]]
```

Out[•]= 4.49958

Out[•]= 4.49055

In[•]:=
```
N[ClosedFormRMFC[15]]
```

Out[•]= 4.49969

In[•]:=
```
Table[{x, Abs[(N[ClosedFormRMFC[10^x]] - N[RMFCApproximation[10^x]])] /
    N[ClosedFormRMFC[10^x], 50]}, {x, 0, 5}]
```

Out[•]=
$\{\{0, 0.0608426\}, \{1, 0.0000779859\}, \{2, 1.14278\times10^{-7}\}, \{3, 1.29004\times10^{-10}\}, \{4, 1.37156\times10^{-13}\}, \{5, 0.\}\}$

In[•]:=
```
Table[{x, Abs[(N[ClosedFormRMFC[10^x]] - N[RMFCApproximation1[10^x]])] /
    N[ClosedFormRMFC[10^x], 50]}, {x, 0, 5}]
```

Out[•]=
$\{\{0, 0.349539\}, \{1, 0.00449172\}, \{2, 0.0000476604\}, \{3, 4.84353\times10^{-7}\}, \{4, 4.87866\times10^{-9}\}, \{5, 4.90019\times10^{-11}\}\}$