Pensieve header: A program to enumerate w-knots.

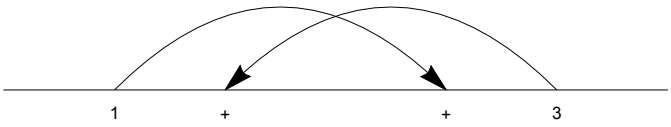
```
SetDirectory["~drorbn/AcademicPensieve/Projects/wEnumeration"]
```

```
/home/drorbn/AcademicPensieve/Projects/wEnumeration
```

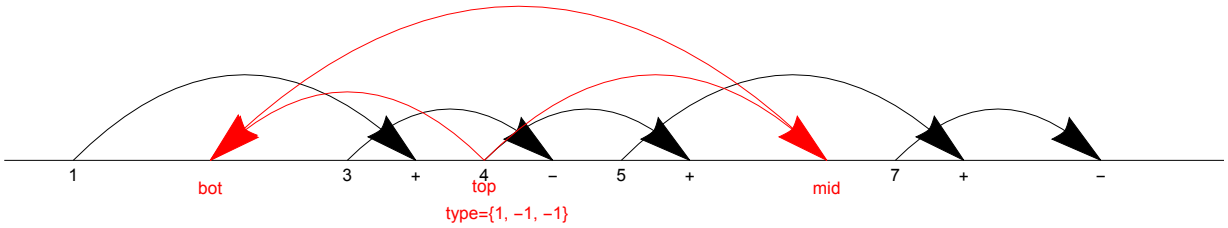
```
A_List \ B_List := Complement[A, B];

Clear[Draw];
DrawArrow[i_, j_] := Arrow[BezierCurve[
    {{Abs@i, 0}, {(Abs@i + Abs@j) / 2, 0.5 Abs[Abs@i - Abs@j]}, {Abs@j, 0}}]];
Draw[wLDiag[ts___Integer] | wCDiag[ts___Integer]] := Module[{n, w, w1},
   n = Length[w = {ts}];
   w1 = Abs /@ w;
   Graphics[{
     Line[{{0, 0}, {n + 1, 0}}],
     Table[If[w[[j]] == 0, {},
       {
        DrawArrow[w1[[j]] - 0.5, j],
        Text[If[w[[j]] > 0, "+", "-"], {j, -0.1}],
        Text[w1[[j]], {w1[[j]] - 0.5, -0.1}]
       }
      ],
      {j, n}
     ]
    }, ImageSize → 480]
  ];
Draw[wLDiag[R3[top_, mid_, bot_, s1_, s2_, s3_], ts__] |
    wCDiag[R3[top_, mid_, bot_, s1_, s2_, s3_], ts__]] := Graphics[Join[
    First@Draw[wLDiag[ts]],
    {Red,
     DrawArrow[top - 0.5, mid],
     DrawArrow[top - 0.5, bot + 0.5],
     DrawArrow[mid, bot + 0.5],
     Text["bot", {bot + 0.5, -0.2}],
     Text["mid", {mid, -0.2}],
     Text["top", {top - 0.5, -0.2}],
     Text["type=" <> ToString@{s1, s2, s3}, {(top + mid + bot) / 3, -0.4}]
    }
   ], ImageSize → 480];
Draw[expr_] := expr /. w_wLDiag | w_wCDiag :> Draw[w]
```

```
Draw[wLDiag[3, 1]]
```




```
Draw[wLDiag[R3[4, 6, 1, 1, -1, -1], 0, 0, 1, -3, 4, 0, 5, -7]]
```




```
AllLinearDiagrams[n_] := Flatten@Table[
   wLDiag @@@ Tuples[Range[k + 1] ⋃ (-Range[k + 1]), k],
   {k, 0, n}
  ]
```

```
AllLinearDiagrams[2]
```

```
{wLDiag[], wLDiag[-2], wLDiag[-1], wLDiag[1], wLDiag[2], wLDiag[-3, -3],
 wLDiag[-3, -2], wLDiag[-3, -1], wLDiag[-3, 1], wLDiag[-3, 2], wLDiag[-3, 3],
 wLDiag[-2, -3], wLDiag[-2, -2], wLDiag[-2, -1], wLDiag[-2, 1], wLDiag[-2, 2],
 wLDiag[-2, 3], wLDiag[-1, -3], wLDiag[-1, -2], wLDiag[-1, -1], wLDiag[-1, 1],
 wLDiag[-1, 2], wLDiag[-1, 3], wLDiag[1, -3], wLDiag[1, -2], wLDiag[1, -1],
 wLDiag[1, 1], wLDiag[1, 2], wLDiag[1, 3], wLDiag[2, -3], wLDiag[2, -2],
 wLDiag[2, -1], wLDiag[2, 1], wLDiag[2, 2], wLDiag[2, 3], wLDiag[3, -3],
 wLDiag[3, -2], wLDiag[3, -1], wLDiag[3, 1], wLDiag[3, 2], wLDiag[3, 3]}
```

```
wCDiag /: RotateLeft[w_wCDiag] := Module[{n},
  n = Length[w];
  wCDiag @@ (RotateLeft[List @@ w] /. j_Integer :> Which[
       j == 1, n,
       j == -1, -n,
       j > 1, j - 1,
       j < -1, j + 1
      ])
 ]
```

```
RotateLeft[wCDiag[-3, 1, 3, -2]]
```

```
wCDiag[4, 2, -1, -2]
```

```
RotateToMinimal[w_wCDiag] := Module[
   {bestw = w, rotatedw = RotateLeft[w]},
   While[rotatedw =!= w,
    bestw = First[Sort[{bestw, rotatedw}]];
    rotatedw = RotateLeft[rotatedw]
   ];
   bestw
  ];
```

```
wDiag[5, 2, -1, -2] // RotateToMinimal
```

```
wDiag[-5, -1, 4, 1]
```

```
wCDiag[w_wLDiag] := Module[{n},
  n = Length[w];
  RotateToMinimal[wCDiag @@ w /. {n + 1 → 1, -n - 1 → -1}]
 ]
```

```
AllCircularDiagrams[n_] := Union[RotateToMinimal /@ Flatten@Table[
     wCDiag @@@ Tuples[Range[k] ⋃ (-Range[k]), k],
     {k, 0, n}
    ]]
```

```
AllCircularDiagrams[2]
```

```
{wCDiag[], wCDiag[-1], wCDiag[1], wCDiag[-2, -2],
 wCDiag[-2, -1], wCDiag[-2, 1], wCDiag[-2, 2], wCDiag[-1, -2],
 wCDiag[-1, 1], wCDiag[-1, 2], wCDiag[1, 1], wCDiag[1, 2], wCDiag[2, 1]}
```

```
RemoveR1[w_wLDiag] := Module[{j, k = 0},
  Do[If[MemberQ[{j, j + 1}, Abs[w[[j]]]], k = j], {j, Length[w]}];
  If[k == 0, w,
   Delete[w, k] /. j_Integer /; Abs[j] > k :> Sign[j] (Abs[j] - 1)
  ]
 ]
```

```
RemoveR1[wLDiag[-4, 1, 3, -4]]
```

```
wLDiag[-4, 1, 3]
```

```
RemoveR1 /@ AllLinearDiagrams[2]
```

```
{wLDiag[], wLDiag[], wLDiag[], wLDiag[], wLDiag[], wLDiag[-2], wLDiag[-2],
 wLDiag[-3, -1], wLDiag[-3, 1], wLDiag[-2], wLDiag[-2], wLDiag[-2], wLDiag[-2],
 wLDiag[-1], wLDiag[1], wLDiag[-2], wLDiag[-2], wLDiag[-1], wLDiag[-1], wLDiag[-1],
 wLDiag[1], wLDiag[-1], wLDiag[-1], wLDiag[1], wLDiag[1], wLDiag[-1], wLDiag[1],
 wLDiag[1], wLDiag[1], wLDiag[2], wLDiag[2], wLDiag[-1], wLDiag[1], wLDiag[2],
 wLDiag[2], wLDiag[2], wLDiag[2], wLDiag[3, -1], wLDiag[3, 1], wLDiag[2], wLDiag[2]}
```

```
RemoveR1[wCDiag[]] = wCDiag[];
RemoveR1[w_wCDiag] := Module[{n, j, k = 0},
  n = Length[w];
  Do[If[MemberQ[{j, j + 1}, Abs[w[[j]]]], k = j], {j, n - 1}];
  If[k != 0,
   Delete[w, k] /. j_Integer /; Abs[j] > k :> Sign[j] (Abs[j] - 1),
   (*else*) If[! MemberQ[{1, n}, Abs[Last[w]]], w,
    Drop[w, -1] /. {n → 1, -n → -1}]
  ]
 ]
```

```
RemoveR1 /@ AllCircularDiagrams[2]
```

```
{wCDiag[], wCDiag[], wCDiag[], wCDiag[-1], wCDiag[-1], wCDiag[1], wCDiag[1],
 wCDiag[-1], wCDiag[1], wCDiag[1], wCDiag[1], wCDiag[1], wCDiag[1]}
```

```
RemoveR1s[w_wLDiag | w_wCDiag] := FixedPoint[RemoveR1, w]
```

```
RemoveR1s /@ AllLinearDiagrams[2] // Union
```

```
{wLDiag[], wLDiag[-3, -1], wLDiag[-3, 1], wLDiag[3, -1], wLDiag[3, 1]}
```

```
RemoveR1s /@ AllCircularDiagrams[4] // Union
```

```
{wCDiag[], wCDiag[-3, -1, -2], wCDiag[-3, -1, 2], wCDiag[-3, 1, -2],
 wCDiag[-3, 1, 2], wCDiag[3, 1, 2], wCDiag[-4, -4, -2, -3], wCDiag[-4, -4, -2, -2],
 wCDiag[-4, -4, -2, 2], wCDiag[-4, -4, -2, 3], wCDiag[-4, -4, -1, -3],
 wCDiag[-4, -4, -1, -2], wCDiag[-4, -4, -1, 2], wCDiag[-4, -4, -1, 3],
 wCDiag[-4, -4, 1, -3], wCDiag[-4, -4, 1, -2], wCDiag[-4, -4, 1, 2],
 wCDiag[-4, -4, 1, 3], wCDiag[-4, -4, 2, -3], wCDiag[-4, -4, 2, -2],
 wCDiag[-4, -4, 2, 2], wCDiag[-4, -4, 2, 3], wCDiag[-4, -1, -2, -3],
 wCDiag[-4, -1, -2, 2], wCDiag[-4, -1, -2, 3], wCDiag[-4, -1, 1, -2],
 wCDiag[-4, -1, 1, 2], wCDiag[-4, -1, 1, 3], wCDiag[-4, -1, 2, -2],
 wCDiag[-4, -1, 2, 2], wCDiag[-4, -1, 2, 3], wCDiag[-4, 1, -2, 2],
 wCDiag[-4, 1, -2, 3], wCDiag[-4, 1, -1, -2], wCDiag[-4, 1, -1, 2],
 wCDiag[-4, 1, -1, 3], wCDiag[-4, 1, 1, -2], wCDiag[-4, 1, 1, 2],
 wCDiag[-4, 1, 1, 3], wCDiag[-4, 1, 2, -2], wCDiag[-4, 1, 2, 2], wCDiag[-4, 1, 2, 3],
 wCDiag[-4, 4, -2, 2], wCDiag[-4, 4, -1, -2], wCDiag[-4, 4, -1, 2],
 wCDiag[-4, 4, -1, 3], wCDiag[-4, 4, 1, -2], wCDiag[-4, 4, 1, 2],
 wCDiag[-4, 4, 1, 3], wCDiag[-4, 4, 2, -2], wCDiag[-4, 4, 2, 2], wCDiag[-4, 4, 2, 3],
 wCDiag[-3, -4, -1, -2], wCDiag[-3, -4, -1, 2], wCDiag[-3, -4, -1, 3],
 wCDiag[-3, -4, 1, 2], wCDiag[-3, -4, 1, 3], wCDiag[-3, -4, 2, 2],
 wCDiag[-3, -4, 2, 3], wCDiag[-3, 1, -1, 2], wCDiag[-3, 1, -1, 3],
 wCDiag[-3, 1, 1, 2], wCDiag[-3, 1, 1, 3], wCDiag[-3, 1, 2, 2], wCDiag[-3, 1, 2, 3],
 wCDiag[-3, 4, -1, 2], wCDiag[-3, 4, 1, 2], wCDiag[-3, 4, 1, 3],
 wCDiag[-3, 4, 2, 2], wCDiag[-3, 4, 2, 3], wCDiag[3, 1, 1, 2], wCDiag[3, 1, 1, 3],
 wCDiag[3, 1, 2, 2], wCDiag[3, 1, 2, 3], wCDiag[3, 4, 1, 2], wCDiag[4, 1, 2, 3]}
```

```
RemoveR2[w_wLDiag] := Module[{j, k = 0},
  Do[
   If[w[[j]] + w[[j + 1]] == 0 && ! MemberQ[Abs[List @@ w], j + 1], k = j], {j, Length[w] - 1}];
  If[k == 0, w,
   Delete[w, {{k}, {k + 1}}] /. j_Integer /; Abs[j] > k :> Sign[j] (Abs[j] - 2)
  ]
 ]
```

```
wLDiag[2, -2] // RemoveR2
```

```
wLDiag[2, -2]
```

```
RemoveR2 /@ AllLinearDiagrams[2]
```

```
{wLDiag[], wLDiag[-2], wLDiag[-1], wLDiag[1], wLDiag[2], wLDiag[-3, -3],
 wLDiag[-3, -2], wLDiag[-3, -1], wLDiag[-3, 1], wLDiag[-3, 2], wLDiag[],
 wLDiag[-2, -3], wLDiag[-2, -2], wLDiag[-2, -1], wLDiag[-2, 1], wLDiag[-2, 2],
 wLDiag[-2, 3], wLDiag[-1, -3], wLDiag[-1, -2], wLDiag[-1, -1], wLDiag[],
 wLDiag[-1, 2], wLDiag[-1, 3], wLDiag[1, -3], wLDiag[1, -2], wLDiag[],
 wLDiag[1, 1], wLDiag[1, 2], wLDiag[1, 3], wLDiag[2, -3], wLDiag[2, -2],
 wLDiag[2, -1], wLDiag[2, 1], wLDiag[2, 2], wLDiag[2, 3], wLDiag[],
 wLDiag[3, -2], wLDiag[3, -1], wLDiag[3, 1], wLDiag[3, 2], wLDiag[3, 3]}
```

```
AllLinearDiagrams[2]
```

```
{wLDiag[], wLDiag[-2], wLDiag[-1], wLDiag[1], wLDiag[2], wLDiag[-3, -3],
 wLDiag[-3, -2], wLDiag[-3, -1], wLDiag[-3, 1], wLDiag[-3, 2], wLDiag[-3, 3],
 wLDiag[-2, -3], wLDiag[-2, -2], wLDiag[-2, -1], wLDiag[-2, 1], wLDiag[-2, 2],
 wLDiag[-2, 3], wLDiag[-1, -3], wLDiag[-1, -2], wLDiag[-1, -1], wLDiag[-1, 1],
 wLDiag[-1, 2], wLDiag[-1, 3], wLDiag[1, -3], wLDiag[1, -2], wLDiag[1, -1],
 wLDiag[1, 1], wLDiag[1, 2], wLDiag[1, 3], wLDiag[2, -3], wLDiag[2, -2],
 wLDiag[2, -1], wLDiag[2, 1], wLDiag[2, 2], wLDiag[2, 3], wLDiag[3, -3],
 wLDiag[3, -2], wLDiag[3, -1], wLDiag[3, 1], wLDiag[3, 2], wLDiag[3, 3]}
```

```
Select[AllLinearDiagrams[2], (# =!= RemoveR2[#]) &]
```

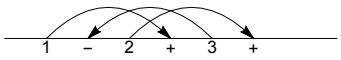
```
{wLDiag[-3, 3], wLDiag[-1, 1], wLDiag[1, -1], wLDiag[3, -3]}
```

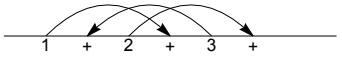
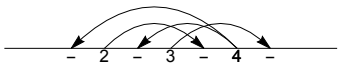
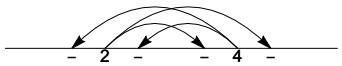
```
RemoveR2[w_wCDiag] /; Length[w] < 2 := w;
RemoveR2[w_wCDiag] := Module[{n, j, k = 0},
  n = Length[w];
  Do[If[w[[j]] + w[[j + 1]] == 0 && ! MemberQ[Abs[List @@ w], j + 1], k = j], {j, n - 1}];
  If[k ≠ 0,
   Delete[w, {{k}, {k + 1}}] /.
     j_Integer /; Abs[j] > k :> Sign[j] (Abs[j] - 2) /. {n - 1 → 1, 1 - n → -1},
   If[w[[1]] + w[[n]] == 0 && ! MemberQ[Abs[List @@ w], 1],
    w[[2 ;; n - 1]] /. j_Integer :> Sign[j] (Abs[j] - 1) /. {n - 1 → 1, 1 - n → -1},
    (*else*) w
   ]
  ]
 ]
```

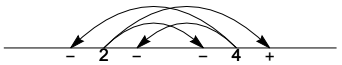
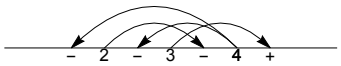
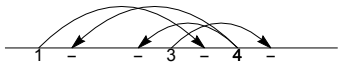
```
RemoveR12s[w_wLDiag | w_wCDiag] := FixedPoint[RemoveR2[RemoveR1[#]] &, w]
```

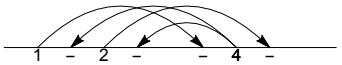
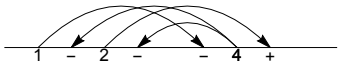
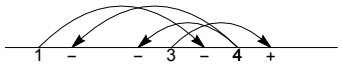
```
Union[RemoveR12s /@ AllCircularDiagrams[4]] // Draw
```

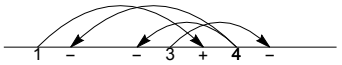
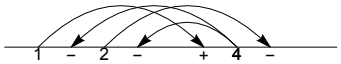
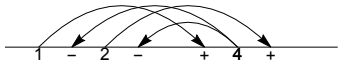
{ ——— , 
1 – 2 – 3 –
, 
1 – 2 – 3 +
,

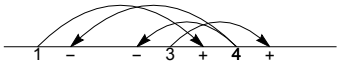
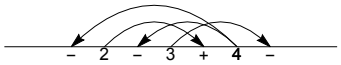
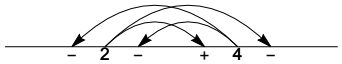

1 – 2 + 3 –
, 
1 – 2 + 3 +
,

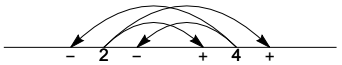
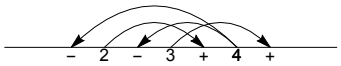
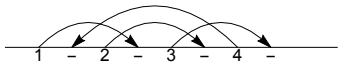

1 + 2 + 3 +
, 
– 2 – 3 – 4 –
, 
– 2 – – 4 –
,

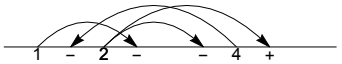
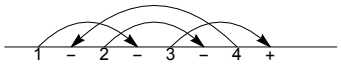
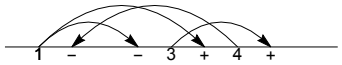

– 2 – – 4 +
, 
– 2 – 3 – 4 +
, 
1 – – 3 – 4 –
,

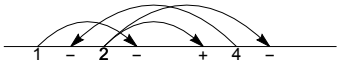
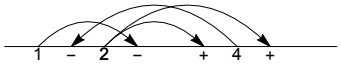
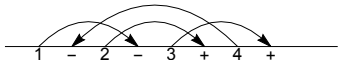

1 – 2 – – 4 –
, 
1 – 2 – – 4 +
, 
1 – – 3 – 4 +
,

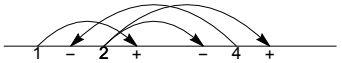
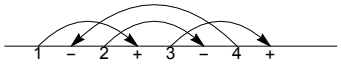
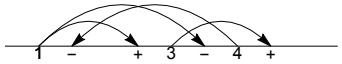

1 – – 3 + 4 –
, 
1 – 2 – + 4 –
, 
1 – 2 – + 4 +
,

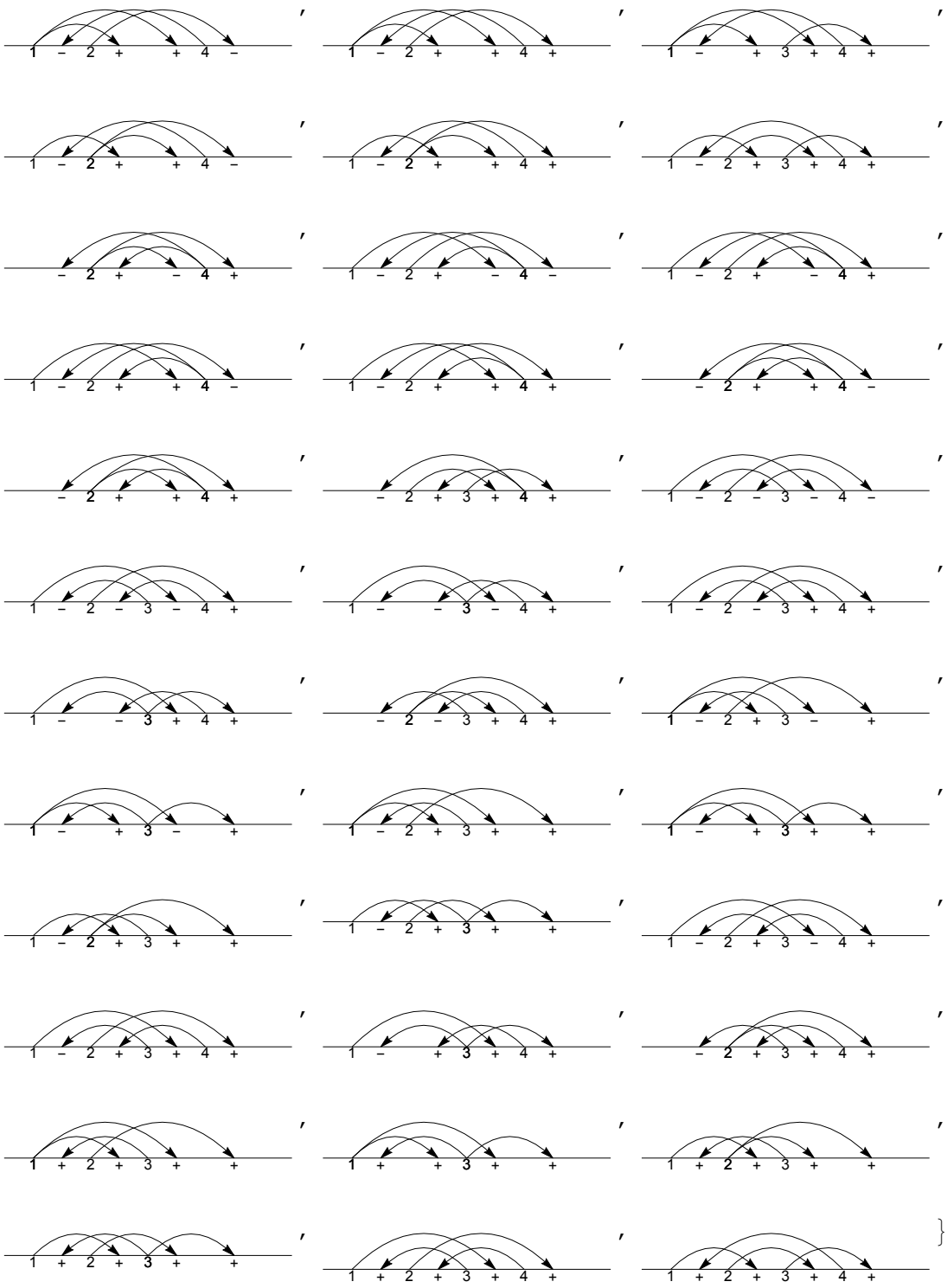

1 – – 3 + 4 +
, 
– 2 – 3 + 4 –
, 
– 2 – + 4 –
,


– 2 – + 4 +
, 
– 2 – 3 + 4 +
, 
1 – 2 – 3 – 4 –
,


1 – 2 – – 4 +
, 
1 – 2 – 3 – 4 +
, 
1 – – 3 + 4 +
,

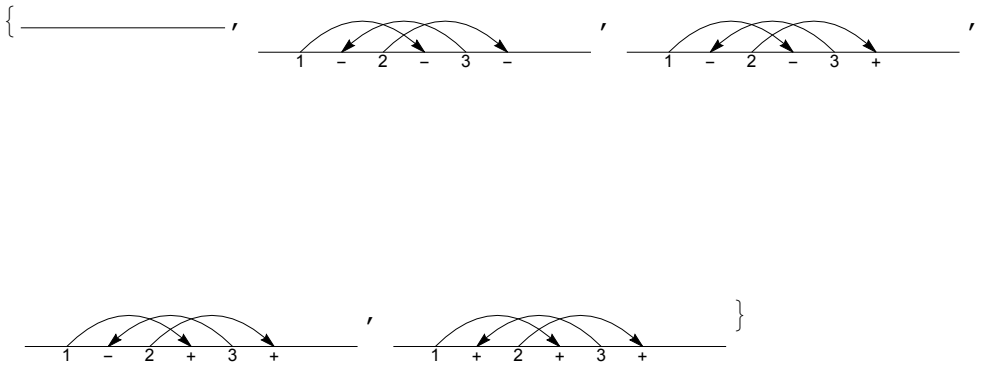

1 – 2 – + 4 –
, 
1 – 2 – + 4 +
, 
1 – 2 – 3 + 4 +
,


1 – 2 + – 4 +
, 
1 – 2 + 3 – 4 +
, 
1 – + 3 – 4 +
,



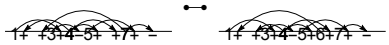
```
RF[w_wCDiag] := RotateToMinimal[RemoveR12s[w]];
RF[w_wLDiag] := RemoveR12s[w];
```

```
Union[RF /@ AllCircularDiagrams[3]] // Draw
```

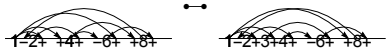



```
wLDiag /:
  Resolve[wLDiag[R3[top_, mid_, bot_, s1_, s2_, s3_], ts__]] := UndirectedEdge[
   RF@ReplacePart[wLDiag@ts, {bot + (1 - s3) / 2 → s2 s3 top,
       bot + (1 + s3) / 2 → s1 s3 (mid + 1), mid → s2 top}],
   RF@ReplacePart[wLDiag@ts, {bot + (1 - s3) / 2 → s1 s3 mid,
       bot + (1 + s3) / 2 → s2 s3 top, mid → s2 top}]
  ];
wCDiag /: Resolve[wCDiag[R3[top_, mid_, bot_, s1_, s2_, s3_], ts__]] :=
 (RF[wCDiag[#]]) & /@ Resolve[wLDiag[R3[top, mid, bot, s1, s2, s3], ts]]

Resolve@wLDiag[R3[4, 6, 1, 1, 1, 1], 0, 0, +1, -3, +4, 0, +5, -7] // Draw
```



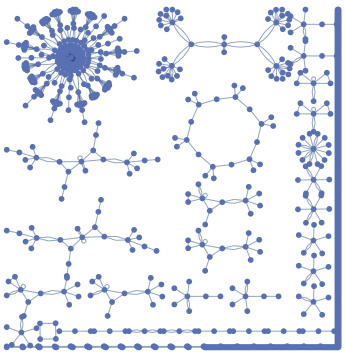
```
Resolve@wCDiag[R3[4, 6, 1, 1, 1, 1], 0, 0, +1, -3, +4, 0, +5, -7] // Draw
```



```
AllLinearR3s[n_] /; n < 3 := {};
AllLinearR3s[n_] := Flatten@Table[
    Prepend[
       ReplacePart[wLDiag @@ Table[0, {n}],
        Thread[Range[n] \ {bot, bot + 1, mid} → #]],
       R3[top, mid, bot, s1, s2, s3]
      ] & /@ Tuples[Range[-n - 1, n + 1] \ {-bot - 1, 0, bot + 1}, n - 3],
    {bot, Range[n - 1]},
    {mid, Range[n] \ {bot, bot + 1}}, {top, Range[n + 1] \ {bot + 1}},
    {s1, {-1, 1}}, {s2, {-1, 1}}, {s3, {-1, 1}}
   ];
AllCircularR3s[n_] /; n < 3 := {};
AllCircularR3s[n_] := Flatten@Table[
    Prepend[
       ReplacePart[wCDiag @@ Table[0, {n}], Thread[Range[n] \ {1, 2, mid} → #]],
       R3[top, mid, 1, s1, s2, s3]
      ] & /@ Tuples[Range[-n, n] \ {-2, 0, 2}, n - 3],
    {mid, Range[n] \ {1, 2}}, {top, Range[n] \ {2}},
    {s1, {-1, 1}}, {s2, {-1, 1}}, {s3, {-1, 1}}
   ];
```

```
Timing[n = 5;
 VS = AllCircularDiagrams[n];
 k = 0; vs = Union[(++k; RF[#]) & /@ VS];
 Print[Length@vs];
 rule = Dispatch[Thread[vs → Range[Length@vs]]];
 ES = AllCircularR3s[n];
 Print[Length@ES];
 es = Union[(++k; Resolve[#] /. rule) & /@ ES];
 g[n] = Graph[Range[Length@vs], es]
]
```

```
1387
6144
{43.867331,
```



```
}
```

```
AllCircularDiagrams[7] // Length
```

```
No more memory available.
Mathematica kernel has shut down.
Try quitting other applications and then retry.
```

```
AllCircularR3s[7] // Length
```

```
4 976 640
```

```
Dynamic[k]
```

```
k
```

```
Timing[n = 7;
 VS = AllCircularDiagrams[n];
 k = 0; vs = Union[(++k; RF[#]) & /@ VS];
 Print[Length@vs];
 rule = Dispatch[Thread[vs → Range[Length@vs]]];
 ES = AllCircularR3s[n];
 Print[Length@ES];
 es = Union[(++k; Resolve[#] /. rule) & /@ ES];
 g[n] = Graph[Range[Length@vs], es];
]
```

```
1 226 539
```

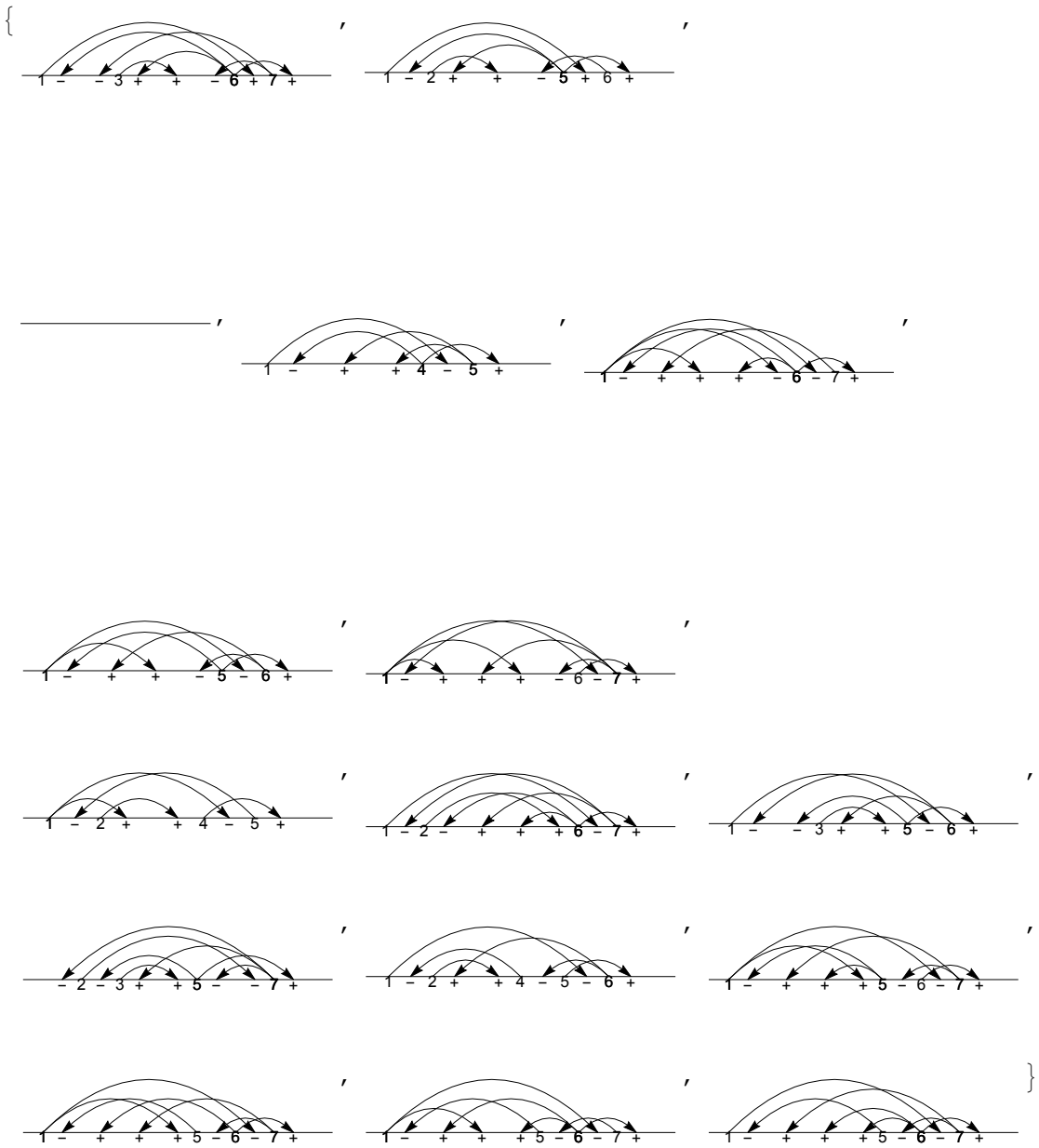
```
4 976 640
```

```
k = 0;
es1 = (++k; Resolve[#] /. rule) & /@ ES;
```

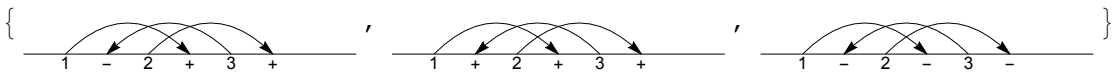
```
cc = ConnectedComponents[g[7]];
Subgraph[g[7], cc[[1]]]
```

```
Subgraph[g[7], cc[[2]]]
```

```
vs[[#]] & /@ FindShortestPath[g[7], cc[[1, 1]], cc[[1, Length@cc[[1]]]]] // Draw
```



```
Select[Table[
   First@MinimalBy[vs[[#]] & /@ c, Length],
   {c, cc}
  ], Length[#] == 3 &] // Draw
```



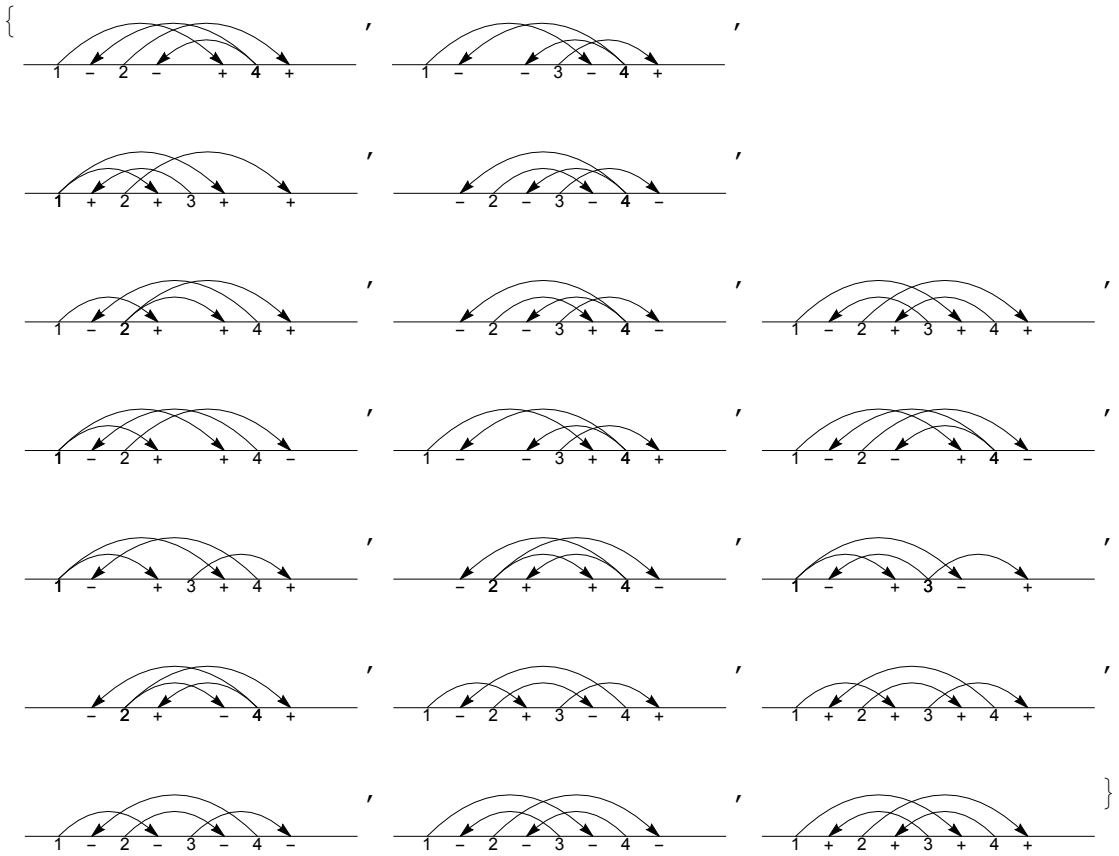
```
Select[Table[
    First@MinimalBy[vs[[#]] & /@ c, Length],
    {c, cc}
  ], Length[#] == 4 &] // Length
```

```
19
```

```
Select[Table[
  First@MinimalBy[vs[[#]] & /@ c, Length],
  {c, cc}
 ], Length[#] == 4 &]
```

```
{wCDiag[-4, -4, 1, 2], wCDiag[-4, -4, -1, 3], wCDiag[3, 1, 1, 2],
 wCDiag[-4, -4, -2, -3], wCDiag[-4, 1, 2, 2], wCDiag[-4, -4, 2, -3],
 wCDiag[-3, 4, 1, 2], wCDiag[-4, 1, 1, -2], wCDiag[-4, -4, 1, 3],
 wCDiag[-4, -4, 1, -2], wCDiag[-4, 1, 1, 3], wCDiag[-4, 4, 2, -2],
 wCDiag[-3, 1, -1, 3], wCDiag[-4, 4, -2, 2], wCDiag[-4, 1, -2, 3], wCDiag[4, 1, 2, 3],
 wCDiag[-4, -1, -2, -3], wCDiag[-3, -4, -1, -2], wCDiag[3, 4, 1, 2]}
```

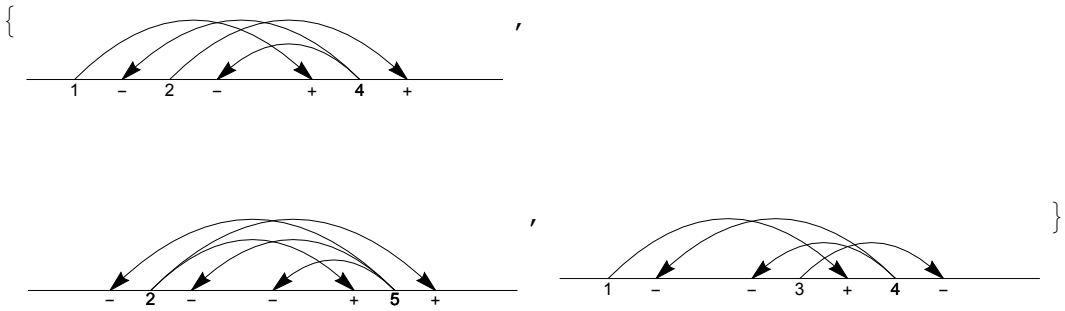
```
{wCDiag[-4, -4, 1, 2], wCDiag[-4, -4, -1, 3], wCDiag[3, 1, 1, 2],
  wCDiag[-4, -4, -2, -3], wCDiag[-4, 1, 2, 2], wCDiag[-4, -4, 2, -3],
  wCDiag[-3, 4, 1, 2], wCDiag[-4, 1, 1, -2], wCDiag[-4, -4, 1, 3],
  wCDiag[-4, -4, 1, -2], wCDiag[-4, 1, 1, 3], wCDiag[-4, 4, 2, -2],
  wCDiag[-3, 1, -1, 3], wCDiag[-4, 4, -2, 2], wCDiag[-4, 1, -2, 3], wCDiag[4, 1, 2, 3],
  wCDiag[-4, -1, -2, -3], wCDiag[-3, -4, -1, -2], wCDiag[3, 4, 1, 2]} // Draw
```



```
{wCDiag[-4, -4, 1, 2], wCDiag[-4, -1, -1, 2] // RF} /. rule
```

```
{16, 14}
```

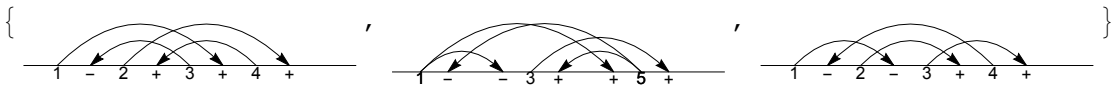
```
vs[[#]] & /@ FindShortestPath[g[7], 16, 14] // Draw
```



```
{wCDiag[-3, 4, 1, 2] // RF, wCDiag[-4, -1, 2, 3] // RF} /. rule
```

```
{59, 28}
```

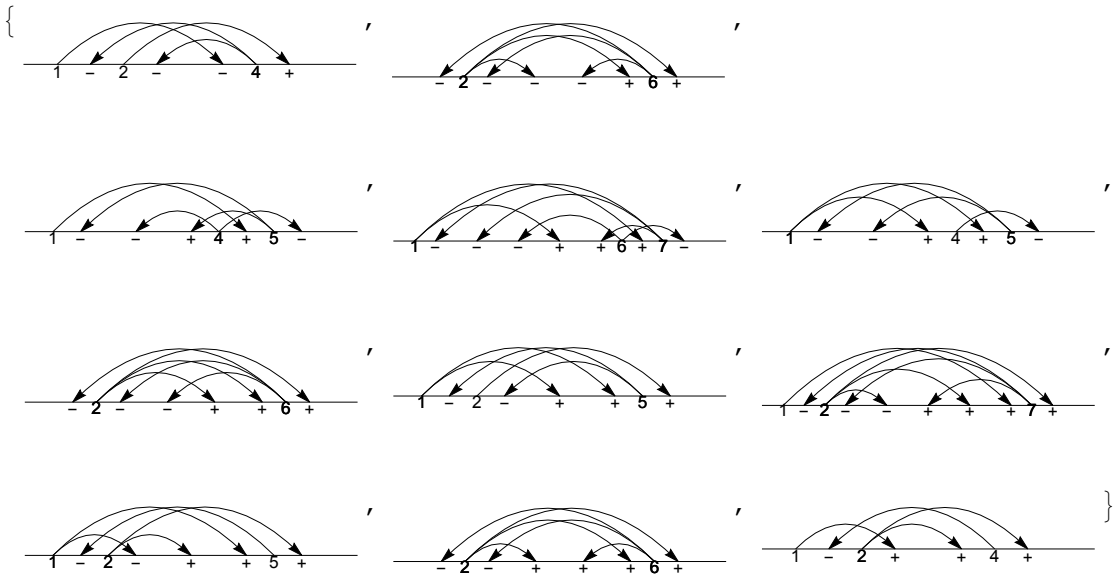
```
vs[[#]] & /@ FindShortestPath[g[7], 59, 28] // Draw
```



```
{wCDiag[-4, -4, -1, 2] // RF, wCDiag[-4, 1, 2, 2] // RF} /. rule
```

```
{12, 36}
```

```
vs[[#]] & /@ FindShortestPath[g[7], 12, 36] // Draw
```



```
{wCDiag[-4, -4, 2, -3] // RF, wCDiag[-3, 1, 1, 2] // RF} /. rule
```

```
{18, 54}
```

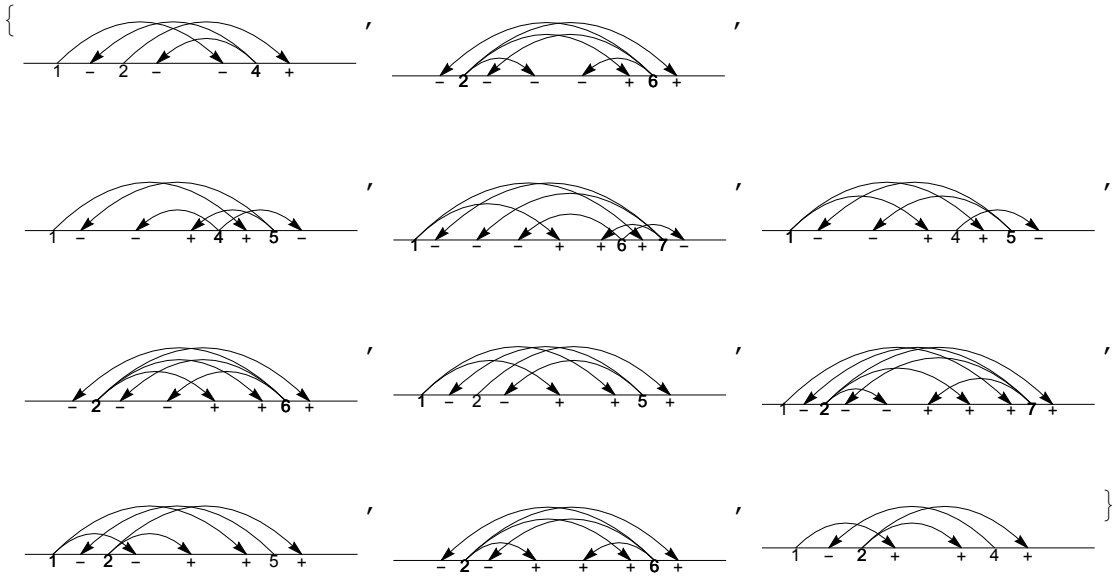
```
vs[[#]] & /@ FindShortestPath[g[7], 12, 54] // Draw
```

```
{}
```

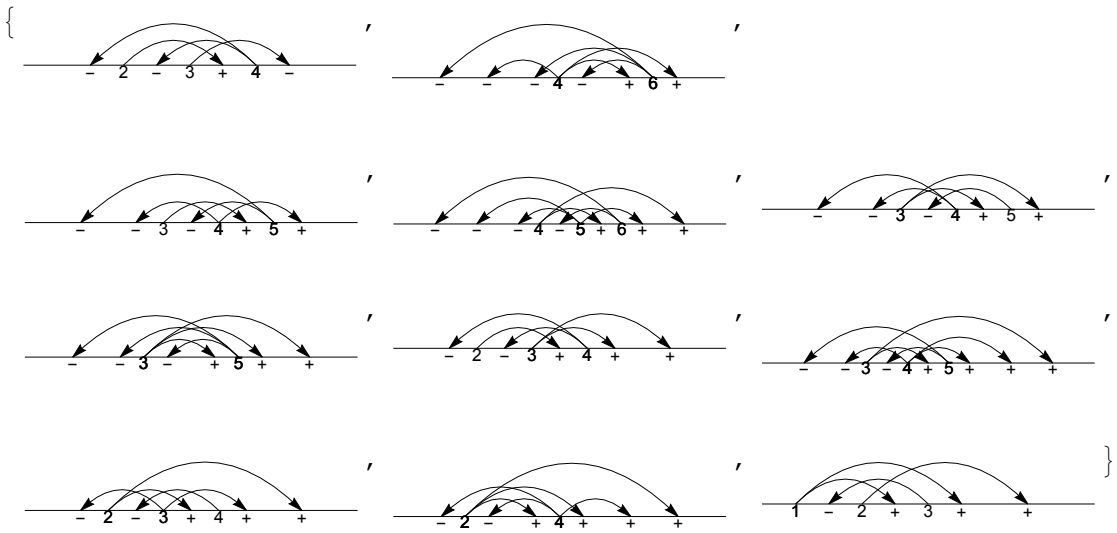
```
los21 = {wCDiag[-4, -4, 1, 2], wCDiag[-3, -4, 2, 2], wCDiag[3, 1, 1, 2],
   wCDiag[-4, -4, -1, -3], wCDiag[-3, 4, 1, 2], wCDiag[-4, -4, -2, 3],
   wCDiag[-4, 1, 1, 3], wCDiag[-4, 4, 2, -2], wCDiag[-4, -4, -1, 2],
   wCDiag[-4, -4, 2, -3], wCDiag[-4, 1, 2, 2], wCDiag[-3, 1, 1, 2],
   wCDiag[-4, -4, 1, 3], wCDiag[-4, 1, 1, -2], wCDiag[-4, 4, -2, 2],
   wCDiag[-3, 1, -1, 3], wCDiag[-4, -1, -2, -3], wCDiag[3, 4, 1, 2],
   wCDiag[-3, -4, -1, -2], wCDiag[4, 1, 2, 3], wCDiag[-3, 4, -1, 2]}
```

```
{wCDiag[-4, -4, 1, 2], wCDiag[-3, -4, 2, 2], wCDiag[3, 1, 1, 2],
 wCDiag[-4, -4, -1, -3], wCDiag[-3, 4, 1, 2], wCDiag[-4, -4, -2, 3],
 wCDiag[-4, 1, 1, 3], wCDiag[-4, 4, 2, -2], wCDiag[-4, -4, -1, 2],
 wCDiag[-4, -4, 2, -3], wCDiag[-4, 1, 2, 2], wCDiag[-3, 1, 1, 2],
 wCDiag[-4, -4, 1, 3], wCDiag[-4, 1, 1, -2], wCDiag[-4, 4, -2, 2],
 wCDiag[-3, 1, -1, 3], wCDiag[-4, -1, -2, -3], wCDiag[3, 4, 1, 2],
 wCDiag[-3, -4, -1, -2], wCDiag[4, 1, 2, 3], wCDiag[-3, 4, -1, 2]}
```

```
findpath[i_, j_] := vs[[#]] & /@
  FindShortestPath[g[7], (los21[[i]] // RF) /. rule, (los21[[j]] // RF) /. rule]
findpath[9, 11] // Draw
```



```
findpath[10, 12] // Draw
```



```
path = FindShortestPath[g[7], (los21[[9]] // RF) /. rule, (los21[[11]] // RF) /. rule]
```

```
{12, 2383, 426, 52 186, 178, 1722, 181, 72 762, 484, 4034, 36}
```

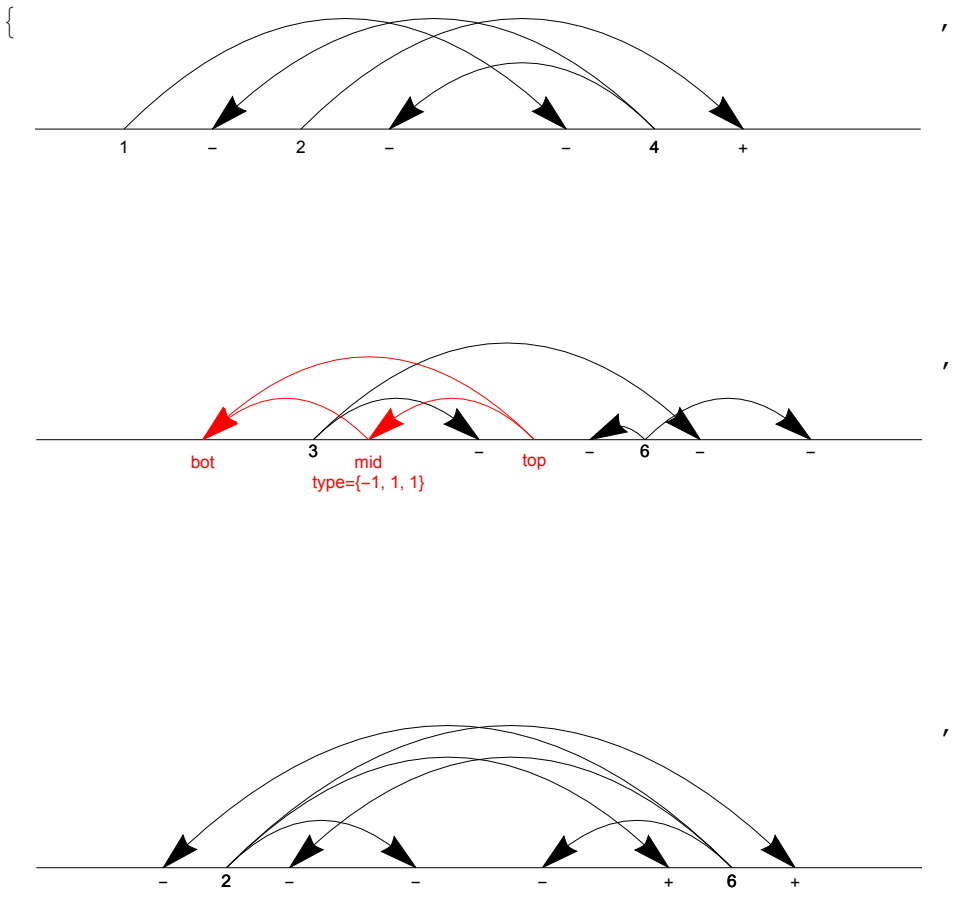
```
R3path = ES[[#]] & /@ (Position[es1, #] | (Reverse@#)][[1, 1]] & /@
    UndirectedEdge @@@ Partition[path, 2, 1])
```

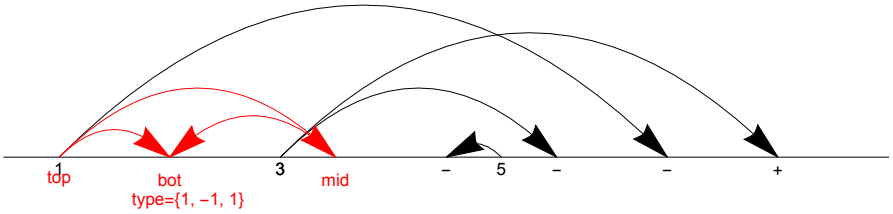
```
{wCDiag[R3[5, 3, 1, -1, 1, 1], 0, 0, 0, -3, -6, -3, -6],
 wCDiag[R3[1, 3, 1, 1, -1, 1], 0, 0, 0, -5, -3, -1, 3],
 wCDiag[R3[4, 5, 1, 1, -1, 1], 0, 0, 5, 6, 0, -5, -5],
 wCDiag[R3[5, 3, 1, 1, 1, 1], 0, 0, 0, -3, -4, -4, -3],
 wCDiag[R3[3, 6, 1, -1, 1, 1], 0, 0, -7, -7, 3, 0, -7],
 wCDiag[R3[1, 3, 1, 1, -1, 1], 0, 0, 0, -7, 3, 3, -7],
 wCDiag[R3[6, 4, 1, 1, 1, 1], 0, 0, 5, 0, -4, -4, -6],
 wCDiag[R3[5, 6, 1, 1, -1, -1], 0, 0, 7, 6, 7, 0, -5],
 wCDiag[R3[3, 6, 1, -1, 1, 1], 0, 0, -7, 3, 1, 0, -7],
 wCDiag[R3[4, 5, 1, -1, -1, -1], 0, 0, 6, 6, 0, -7, -4]}
```

```
Riffle[vs[[#]] & /@ path, R3path]
```

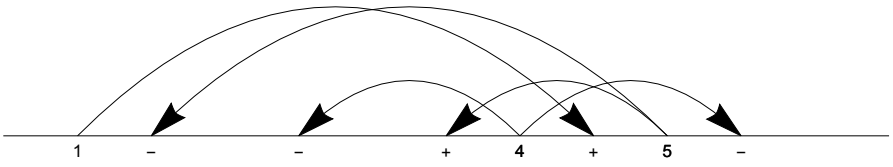
```
{wCDiag[-4, -4, -1, 2],
 wCDiag[R3[5, 3, 1, -1, 1, 1], 0, 0, 0, -3, -6, -3, -6], wCDiag[-6, -6, -2, -6, 2, 2],
 wCDiag[R3[1, 3, 1, 1, -1, 1], 0, 0, 0, -5, -3, -1, 3], wCDiag[-5, -4, 5, 1, -4],
 wCDiag[R3[4, 5, 1, 1, -1, 1], 0, 0, 5, 6, 0, -5, -5], wCDiag[-7, -7, -6, 1, 7, 1, -6],
 wCDiag[R3[5, 3, 1, 1, 1, 1], 0, 0, 0, -3, -4, -4, -3], wCDiag[-5, -5, 1, 1, -4],
 wCDiag[R3[3, 6, 1, -1, 1, 1], 0, 0, -7, -7, 3, 0, -7], wCDiag[-6, -6, -6, 2, 2, 2],
 wCDiag[R3[1, 3, 1, 1, -1, 1], 0, 0, 0, -7, 3, 3, -7], wCDiag[-5, -5, 1, 1, 2],
 wCDiag[R3[6, 4, 1, 1, 1, 1], 0, 0, 5, 0, -4, -4, -6], wCDiag[-7, -7, -2, 7, 2, 1, 2],
 wCDiag[R3[5, 6, 1, 1, -1, -1], 0, 0, 7, 6, 7, 0, -5], wCDiag[-5, -1, 2, 1, 2],
 wCDiag[R3[3, 6, 1, -1, 1, 1], 0, 0, -7, 3, 1, 0, -7], wCDiag[-6, -6, 2, 6, 2, 2],
 wCDiag[R3[4, 5, 1, -1, -1, -1], 0, 0, 6, 6, 0, -7, -4], wCDiag[-4, 1, 2, 2]}
```

```
Riffle[vs[[#]] & /@ path, R3path] // Draw
```

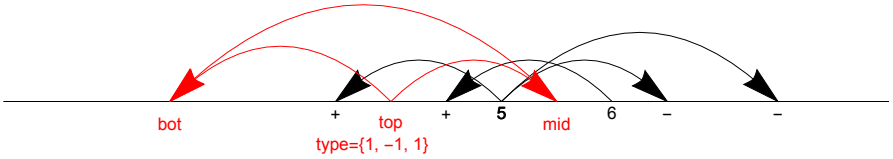



1
top
bot
type={1, −1, 1}
3
mid
−
5
−
−
+

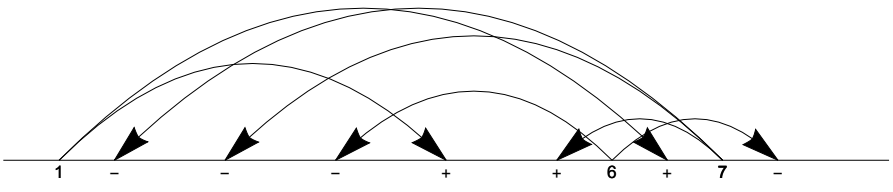
,


1
−
−
+
4
+
5
−

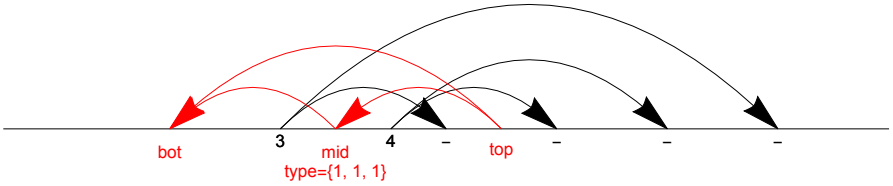
,


bot
+
top
+
5
mid
6
−
−
type={1, −1, 1}

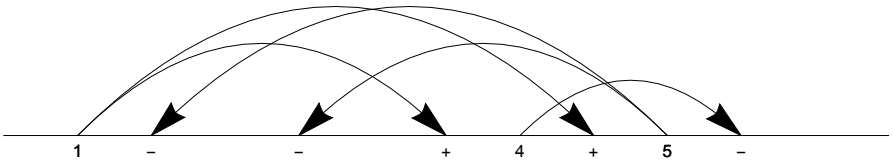
,


1
−
−
−
+
+
6
+
7
−

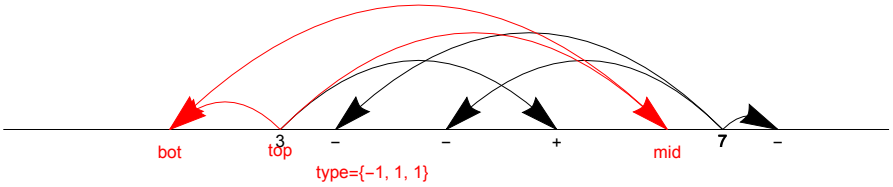
,


bot
3
mid
4
top
type={1, 1, 1}

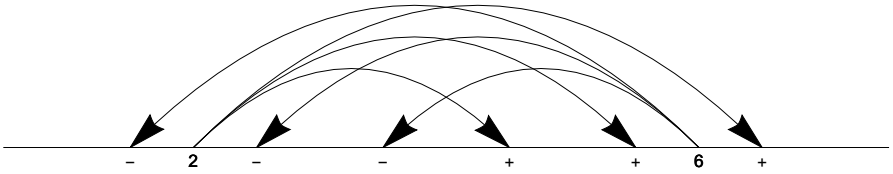
,


1
4
5

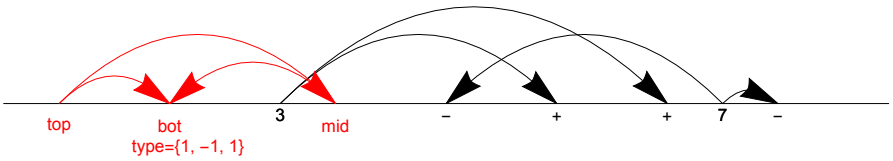
,


bot
3
top
mid
7
type={−1, 1, 1}

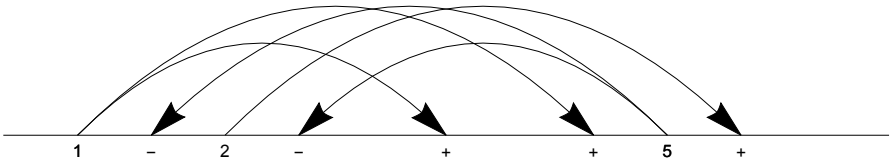
,


2
6

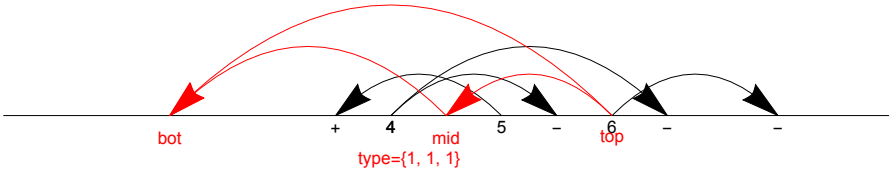
,


top
bot
type={1, −1, 1}
3
mid
−
+
+
7
−

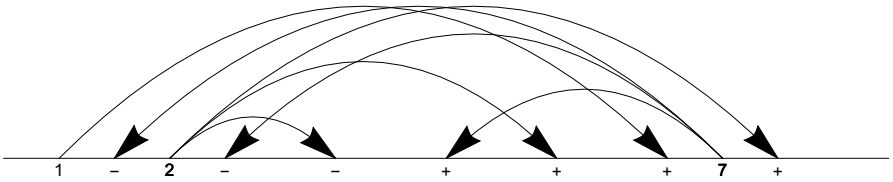
,


1
−
2
−
+
+
5
+

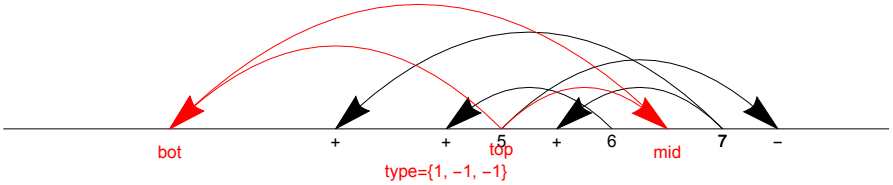
,


bot
+
4
mid
type={1, 1, 1}
5
−
6
top
−
−

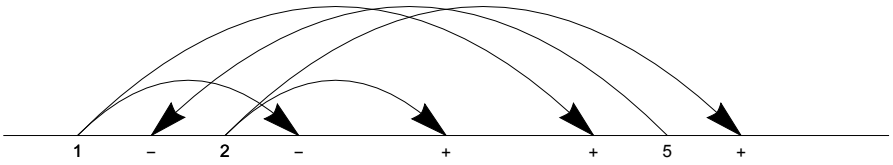
,


1
−
2
−
−
+
+
+
7
+

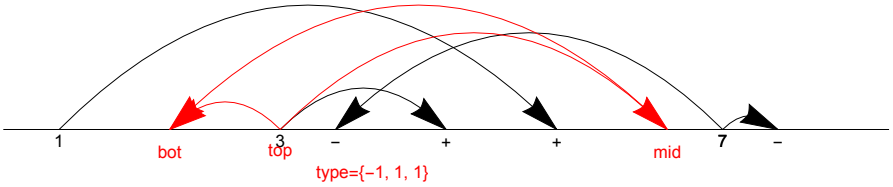
,


bot
+
+
5
top
+
6
7
mid
−
type={1, −1, −1}

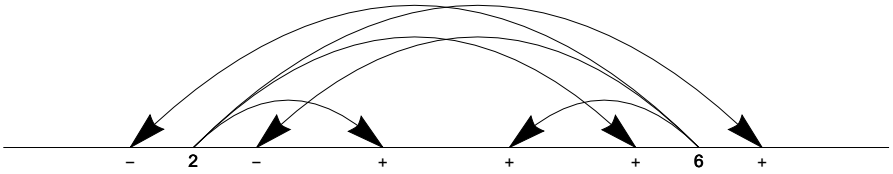
,

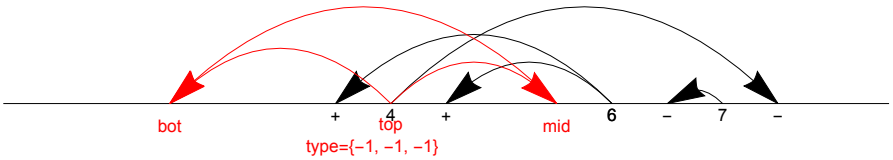

1
−
2
−
+
+
5
+

,

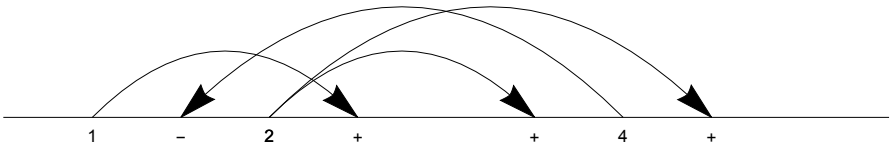

1
bot
3
top
−
+
+
mid
7
−
type={−1, 1, 1}

,


−
2
−
+
+
+
6
+


bot
+
4
top
+
mid
6
–
7
–
type={–1, –1, –1}

,


1
–
2
+
+
4
+

}