In[•]:=
```
Directory[]
```

Out[•]=
```
C:\Users\alber\OneDrive\Documents
```

In[•]:=
```
<< KnotTheory`
```

```
Loading KnotTheory` version of February 2, 2020, 10:53:45.2097.
Read more at http://katlas.org/wiki/KnotTheory.
```

- We will first start by starting with the most straightforward starting point and go from there:
- To do that, because of the massive amount of branching that might happen, I reckon we use recursion:
- The following only works for non-initial and non-final crossings

In[•]:=
```
nextStepSelection[curFronts_, curCrossingOrders_, pd_PD] :=
 Module[{pdList, pdListLeft, listPossibleCrossings,
   listAllowedCrossings, listAfterFront, listAfterCrossings,
   possibilityIndex, numPossibilities, newFronts, newCrossingOrders},
  pdList = List @@ List @@@ pd;
  numPossibilities = Length[curFronts];
  possibilityIndex = 1;
  newFronts = {};
  newCrossingOrders = {};
  While[possibilityIndex ≤ numPossibilities,
   pdListLeft = Complement[pdList, curCrossingOrders[[possibilityIndex]]];
   listPossibleCrossings =
    Select[pdListLeft, Abs[Length[curFronts[[possibilityIndex]]] - Length[Complement[
           curFronts[[possibilityIndex]] ⋃ #, curFronts[[possibilityIndex]] ⋂ #]]] ≤ 2 &];
   listAllowedCrossings = Select[listPossibleCrossings,
     combineFrontAndCrossing[curFronts[[possibilityIndex]], #] ≠ {-1} &];
   listAfterFront = combineFrontAndCrossing[curFronts[[possibilityIndex]], #] & /@
     listAllowedCrossings;
   listAfterCrossings = Join[curCrossingOrders[[possibilityIndex]], {#}] & /@
     listAllowedCrossings;
   newFronts = Join[newFronts, {listAfterFront}];
   newCrossingOrders = Join[newCrossingOrders, {listAfterCrossings}];
   possibilityIndex = possibilityIndex + 1;
  ];
  If[newFronts == {}, -1, {Flatten[newFronts, 1], Flatten[newCrossingOrders, 1]}]]
```

```
In[•]:= combineFrontAndCrossing[curFront_, curCrossing_] :=
 Module[{intersection, intersectionFrontIndex, intersectionCrossingIndex,
   intersectionFrontIndexDiff, intersectionCrossingIndexDiff, newFront},
  intersection = SortBy[Select[curCrossing, MemberQ[curFront, #] &],
    First[First[Position[curFront, #]]] &];
  intersectionFrontIndex = First[First[Position[curFront, #]]] & /@ intersection;
  intersectionCrossingIndex = First[First[Position[curCrossing, #]]] & /@ intersection;
  intersectionFrontIndexDiff =
   Table[intersectionFrontIndex[[i + 1]] - intersectionFrontIndex[[i]],
    {i, Length[intersectionFrontIndex] - 1}];
  intersectionCrossingIndexDiff =
   Table[intersectionCrossingIndex[[i + 1]] - intersectionCrossingIndex[[i]],
     {i, Length[intersectionCrossingIndex] - 1}] /. -3 → 1;
  newFront = If[(DeleteDuplicates[intersectionFrontIndexDiff] == {1} ||
       DeleteDuplicates[intersectionFrontIndexDiff] == {}) &&
     (DeleteDuplicates[intersectionCrossingIndexDiff] == {1} ||
       DeleteDuplicates[intersectionCrossingIndexDiff] == {}),
    Join[curFront[[1 ;; intersectionFrontIndex[[1]] - 1]],
     orderUnconnected[curCrossing, intersectionCrossingIndex],
     curFront[[intersectionFrontIndex[[-1]] + 1 ;; Length[curFront]]]], {-1}];
  newFront]

In[•]:= orderUnconnected[crossing_, connected_] :=
 Module[{containOne, order},
  containOne = MemberQ[connected, 1];
  order = If[containOne,
    Reverse[crossing[[connected[[-1]] + 1 ;; If[connected[[1]] == 1, 4, connected[[1]] - 1]]]],
    Join[Reverse[crossing[[1 ;; connected[[1]] - 1]]],
     Reverse[crossing[[connected[[-1]] + 1 ;; 4]]]]];
  order]
```

- Given the initial line segment stacking order we now try to connect the rest of the line segments

```
In[•]:= quickCrossingOrderGivenStart[pd_PD, startCrossing_] :=
 Module[{currentFronts, currentCrossingOrders, index, newCrossingFrontOrders,
   newFronts, newCrossingOrders, newFrontsFinal, newCrossingOrdersFinal},
  index = 2;
  currentFronts =
   {{startCrossing[[3]], startCrossing[[2]], startCrossing[[1]], startCrossing[[4]]}};
  currentCrossingOrders = {selectAppropriateCrossing[pd, startCrossing]};
  While[index < Length[List @@ pd] && IntegerQ[currentFronts] == False,
   newCrossingFrontOrders =
    nextStepSelection[currentFronts, currentCrossingOrders, pd];
   currentFronts = If[IntegerQ[newCrossingFrontOrders], -1, newCrossingFrontOrders[[1]]];
   currentCrossingOrders =
    If[IntegerQ[newCrossingFrontOrders], -1, newCrossingFrontOrders[[2]]];
   index = index + 1;
  ];
  newFrontsFinal = If[IntegerQ[currentFronts], -1,
    Flatten[Table[combineFrontAndCrossing[currentFronts[[i]], First[Complement[List @@
          List @@@ pd, currentCrossingOrders[[i]]]]], {i, Length[currentFronts]}]]];
  newCrossingOrdersFinal = If[IntegerQ[newFrontsFinal], -1,
    Join[#, Complement[List @@ List @@@ pd, #]] & /@ currentCrossingOrders];
  newCrossingOrdersFinal]

In[•]:= selectAppropriateCrossing[pd_PD, givenCrossingOrder_] :=
 Select[List @@ List @@@ pd, ContainsAll[#, givenCrossingOrder] &]
```

- And finally we use these functions to find the minimum q-width for each knot:

```
In[•]:= generateStartingPoints[pd_PD] :=
 Module[{index, indexedCrossing, indexedCrossingOrder, possibleStartingPoints},
  index = 1;
  possibleStartingPoints = {};
  While[index ≤ Length[List @@ pd],
   indexedCrossingOrder = (List @@ List @@@ pd)[[index]];
   possibleStartingPoints = Join[possibleStartingPoints,
     NestList[Join[{#[[-1]]}, Delete[#, -1]] &, indexedCrossingOrder, 3]];
   index = index + 1;
  ];
  possibleStartingPoints]
```

```
In[•]:= quickWidth[pd_PD] := Module[{listCrossings, startCrossingIndex,
          startingCrossings, quickCrossingPermutations, quickCrossingPermutationWidths},
        listCrossings = List @@ List @@@ pd;
        startCrossingIndex = 1;
        startingCrossings = generateStartingPoints [pd];
        quickCrossingPermutations =
         Select[DeleteDuplicates[Flatten[Table[quickCrossingOrderGivenStart[
              pd, startingCrossings[[i]]], {i, Length[startingCrossings]}]
            , 1]], IntegerQ[#] == False &];
        quickCrossingPermutationWidths =
         Min [Max /@ ((Length /@ FoldList[Complement[#1 ⋃ #2, #1 ⋂ #2] &, {}, #]) & /@
             quickCrossingPermutations)];
        quickCrossingPermutationWidths]

In[•]:= quantitiesGivenCrossingNum = <|3 → 1,
        4 → 1, 5 → 2, 6 → 3, 7 → 7, 8 → 21, 9 → 49, 10 → 165|>

Out[•]= <|3 → 1, 4 → 1, 5 → 2, 6 → 3, 7 → 7, 8 → 21, 9 → 49, 10 → 165|>

In[•]:= Table[Echo[Knot[i, j] → quickWidth[PD[Knot[i, j]]]],
         {i, 3, 10}, {j, 1, quantitiesGivenCrossingNum[i]}]

» Knot[3, 1] → 4
» Knot[4, 1] → 4
» Knot[5, 1] → 4
» Knot[5, 2] → 4
» Knot[6, 1] → 4
» Knot[6, 2] → 4
» Knot[6, 3] → 4
» Knot[7, 1] → 4
» Knot[7, 2] → 4
» Knot[7, 3] → 4
» Knot[7, 4] → 4
» Knot[7, 5] → 4
» Knot[7, 6] → 4
» Knot[7, 7] → 4
» Knot[8, 1] → 4
» Knot[8, 2] → 4
» Knot[8, 3] → 4
» Knot[8, 4] → 4
» Knot[8, 5] → 6
» Knot[8, 6] → 4
» Knot[8, 7] → 4
```

» Knot[8, 8] → 4

» Knot[8, 9] → 4

» Knot[8, 10] → 6

» Knot[8, 11] → 4

» Knot[8, 12] → 4

» Knot[8, 13] → 4

» Knot[8, 14] → 4

» Knot[8, 15] → 6

» Knot[8, 16] → 6

» Knot[8, 17] → 6

» Knot[8, 18] → 6

» Knot[8, 19] → 6

» Knot[8, 20] → 6

» Knot[8, 21] → 6

» Knot[9, 1] → 4

» Knot[9, 2] → 4

» Knot[9, 3] → 4

» Knot[9, 4] → 4

» Knot[9, 5] → 4

» Knot[9, 6] → 4

» Knot[9, 7] → 4

» Knot[9, 8] → 4

» Knot[9, 9] → 4

» Knot[9, 10] → 4

» Knot[9, 11] → 4

» Knot[9, 12] → 4

» Knot[9, 13] → 4

» Knot[9, 14] → 4

» Knot[9, 15] → 4

» Knot[9, 16] → 6

» Knot[9, 17] → 4

» Knot[9, 18] → 6

» Knot[9, 19] → 4

» Knot[9, 20] → 4

» Knot[9, 21] → 4

» Knot[9, 22] → 6

» Knot[9, 23] → 4

» Knot[9, 24] → 6

» Knot[9, 25] → 6

» Knot[9, 26] → 4

» Knot[9, 27] → 6

» Knot[9, 28] → 6

» Knot[9, 29] → 6

» Knot[9, 30] → 6

» Knot[9, 31] → 4

» Knot[9, 32] → 6

» Knot[9, 33] → 6

» Knot[9, 34] → 6

» Knot[9, 35] → 6

» Knot[9, 36] → 6

» Knot[9, 37] → 6

» Knot[9, 38] → 6

» Knot[9, 39] → 6

» Knot[9, 40] → 6

» Knot[9, 41] → 6

» Knot[9, 42] → 6

» Knot[9, 43] → 6

» Knot[9, 44] → 6

» Knot[9, 45] → 6

» Knot[9, 46] → 6

» Knot[9, 47] → 6

» Knot[9, 48] → 6

» Knot[9, 49] → 6

» Knot[10, 1] → 4

» Knot[10, 2] → 4

» Knot[10, 3] → 4

» Knot[10, 4] → 4

» Knot[10, 5] → 4

» Knot[10, 6] → 4

» Knot[10, 7] → 4

» Knot[10, 8] → 4

» Knot[10, 9] → 4

» Knot[10, 10] → 4

» Knot[10, 11] → 4

» Knot[10, 12] → 4

» Knot[10, 13] → 4

» Knot[10, 14] → 4

» Knot[10, 15] → 4

» Knot[10, 16] → 4

» Knot[10, 17] → 4

» Knot[10, 18] → 4

» Knot[10, 19] → 4

» Knot[10, 20] → 4

» Knot[10, 21] → 4

» Knot[10, 22] → 4

» Knot[10, 23] → 4

» Knot[10, 24] → 6

» Knot[10, 25] → 6

» Knot[10, 26] → 4

» Knot[10, 27] → 4

» Knot[10, 28] → 4

» Knot[10, 29] → 4

» Knot[10, 30] → 4

» Knot[10, 31] → 4

» Knot[10, 32] → 4

» Knot[10, 33] → 4

» Knot[10, 34] → 4

» Knot[10, 35] → 4

» Knot[10, 36] → 4

» Knot[10, 37] → 4

» Knot[10, 38] → 4

» Knot[10, 39] → 4

» Knot[10, 40] → 4

» Knot[10, 41] → 4

» Knot[10, 42] → 6

» Knot[10, 43] → 4

» Knot[10, 44] → 6

» Knot[10, 45] → 4

» Knot[10, 46] → 6

» Knot[10, 47] → 6

» Knot[10, 48] → 6

» Knot[10, 49] → 6

» Knot[10, 50] → 6

» Knot[10, 51] → 6

» Knot[10, 52] → 6

» Knot[10, 53] → 6

» Knot[10, 54] → 6

» Knot[10, 55] → 6

» Knot[10, 56] → 6

» Knot[10, 57] → 6

» Knot[10, 58] → 6

» Knot[10, 59] → 6

» Knot[10, 60] → 6

» Knot[10, 61] → 6

» Knot[10, 62] → 6

» Knot[10, 63] → 6

» Knot[10, 64] → 6

» Knot[10, 65] → 6

» Knot[10, 66] → 6

» Knot[10, 67] → 6

» Knot[10, 68] → 6

» Knot[10, 69] → 6

» Knot[10, 70] → 6

» Knot[10, 71] → 6

» Knot[10, 72] → 6

» Knot[10, 73] → 6

» Knot[10, 74] → 6

» Knot[10, 75] → 6

» Knot[10, 76] → 6

» Knot[10, 77] → 6

» Knot[10, 78] → 6

» Knot[10, 79] → 6

» Knot[10, 80] → 6

» Knot[10, 81] → 6

» Knot[10, 82] → 6

» Knot[10, 83] → 6

» Knot[10, 84] → 6

» Knot[10, 85] → 6

» Knot[10, 86] → 6

» Knot[10, 87] → 6

» Knot[10, 88] → 6

» Knot[10, 89] → 6

» Knot[10, 90] → 6

» Knot[10, 91] → 6

» Knot[10, 92] → 6

» Knot[10, 93] → 6

» Knot[10, 94] → 6

» Knot[10, 95] → 6

» Knot[10, 96] → 6

» Knot[10, 97] → 6

» Knot[10, 98] → 6

» Knot[10, 99] → 6

» Knot[10, 100] → 6

» Knot[10, 101] → 6

» Knot[10, 102] → 6

» Knot[10, 103] → 6

» Knot[10, 104] → 6

» Knot[10, 105] → 6

» Knot[10, 106] → 6

» Knot[10, 107] → 6

» Knot[10, 108] → 6

» Knot[10, 109] → 6

» Knot[10, 110] → 6

» Knot[10, 111] → 6

» Knot[10, 112] → 6

» Knot[10, 113] → 6

» Knot[10, 114] → 6

» Knot[10, 115] → 6

» Knot[10, 116] → 6

» Knot[10, 117] → 6

» Knot[10, 118] → 6

» Knot[10, 119] → 6

» Knot[10, 120] → 6

» Knot[10, 121] → 6

» Knot[10, 122] → 6

» Knot[10, 123] → 6

» Knot[10, 124] → 6

» Knot[10, 125] → 6

» Knot[10, 126] → 6

» Knot[10, 127] → 6

» Knot[10, 128] → 6

» Knot[10, 129] → 6

» Knot[10, 130] → 6

» Knot[10, 131] → 6

» Knot[10, 132] → 6

» Knot[10, 133] → 6

» Knot[10, 134] → 6

» Knot[10, 135] → 6

» Knot[10, 136] → 6

» Knot[10, 137] → 6

» Knot[10, 138] → 6

» Knot[10, 139] → 6

» Knot[10, 140] → 6

» Knot[10, 141] → 6

» Knot[10, 142] → 6

» Knot[10, 143] → 6

» Knot[10, 144] → 6

» Knot[10, 145] → 6

» Knot[10, 146] → 6

» Knot[10, 147] → 6

» Knot[10, 148] → 6

» Knot[10, 149] → 6

» Knot[10, 150] → 6

» Knot[10, 151] → 6

» Knot[10, 152] → 6

» Knot[10, 153] → 6

```
» Knot[10, 154] → 6
» Knot[10, 155] → 6
» Knot[10, 156] → 6
» Knot[10, 157] → 6
» Knot[10, 158] → 6
» Knot[10, 159] → 6
» Knot[10, 160] → 6
» Knot[10, 161] → 6
» Knot[10, 162] → 6
» Knot[10, 163] → 6
» Knot[10, 164] → 6
» Knot[10, 165] → 6
```

Out[•]=
```
{{Knot[3, 1] → 4}, {Knot[4, 1] → 4}, {Knot[5, 1] → 4, Knot[5, 2] → 4},
 {Knot[6, 1] → 4, Knot[6, 2] → 4, Knot[6, 3] → 4}, {Knot[7, 1] → 4, Knot[7, 2] → 4,
  Knot[7, 3] → 4, Knot[7, 4] → 4, Knot[7, 5] → 4, Knot[7, 6] → 4, Knot[7, 7] → 4},
 {Knot[8, 1] → 4, Knot[8, 2] → 4, Knot[8, 3] → 4, Knot[8, 4] → 4, Knot[8, 5] → 6,
  Knot[8, 6] → 4, Knot[8, 7] → 4, Knot[8, 8] → 4, Knot[8, 9] → 4,
  Knot[8, 10] → 6, Knot[8, 11] → 4, Knot[8, 12] → 4, Knot[8, 13] → 4,
  Knot[8, 14] → 4, Knot[8, 15] → 6, Knot[8, 16] → 6, Knot[8, 17] → 6,
  Knot[8, 18] → 6, Knot[8, 19] → 6, Knot[8, 20] → 6, Knot[8, 21] → 6},
 {Knot[9, 1] → 4, Knot[9, 2] → 4, Knot[9, 3] → 4, Knot[9, 4] → 4, Knot[9, 5] → 4,
  Knot[9, 6] → 4, Knot[9, 7] → 4, Knot[9, 8] → 4, Knot[9, 9] → 4, Knot[9, 10] → 4,
  Knot[9, 11] → 4, Knot[9, 12] → 4, Knot[9, 13] → 4, Knot[9, 14] → 4, Knot[9, 15] → 4,
  Knot[9, 16] → 6, Knot[9, 17] → 4, Knot[9, 18] → 6, Knot[9, 19] → 4, Knot[9, 20] → 4,
  Knot[9, 21] → 4, Knot[9, 22] → 6, Knot[9, 23] → 4, Knot[9, 24] → 6, Knot[9, 25] → 6,
  Knot[9, 26] → 4, Knot[9, 27] → 6, Knot[9, 28] → 6, Knot[9, 29] → 6, Knot[9, 30] → 6,
  Knot[9, 31] → 4, Knot[9, 32] → 6, Knot[9, 33] → 6, Knot[9, 34] → 6, Knot[9, 35] → 6,
  Knot[9, 36] → 6, Knot[9, 37] → 6, Knot[9, 38] → 6, Knot[9, 39] → 6, Knot[9, 40] → 6,
  Knot[9, 41] → 6, Knot[9, 42] → 6, Knot[9, 43] → 6, Knot[9, 44] → 6, Knot[9, 45] → 6,
  Knot[9, 46] → 6, Knot[9, 47] → 6, Knot[9, 48] → 6, Knot[9, 49] → 6},
 {Knot[10, 1] → 4, Knot[10, 2] → 4, Knot[10, 3] → 4, Knot[10, 4] → 4, Knot[10, 5] → 4,
  Knot[10, 6] → 4, Knot[10, 7] → 4, Knot[10, 8] → 4, Knot[10, 9] → 4, Knot[10, 10] → 4,
  Knot[10, 11] → 4, Knot[10, 12] → 4, Knot[10, 13] → 4, Knot[10, 14] → 4, Knot[10, 15] → 4,
  Knot[10, 16] → 4, Knot[10, 17] → 4, Knot[10, 18] → 4, Knot[10, 19] → 4, Knot[10, 20] → 4,
  Knot[10, 21] → 4, Knot[10, 22] → 4, Knot[10, 23] → 4, Knot[10, 24] → 6, Knot[10, 25] → 6,
  Knot[10, 26] → 4, Knot[10, 27] → 4, Knot[10, 28] → 4, Knot[10, 29] → 4, Knot[10, 30] → 4,
  Knot[10, 31] → 4, Knot[10, 32] → 4, Knot[10, 33] → 4, Knot[10, 34] → 4, Knot[10, 35] → 4,
  Knot[10, 36] → 4, Knot[10, 37] → 4, Knot[10, 38] → 4, Knot[10, 39] → 4, Knot[10, 40] → 4,
  Knot[10, 41] → 4, Knot[10, 42] → 6, Knot[10, 43] → 4, Knot[10, 44] → 6, Knot[10, 45] → 4,
  Knot[10, 46] → 6, Knot[10, 47] → 6, Knot[10, 48] → 6, Knot[10, 49] → 6, Knot[10, 50] → 6,
  Knot[10, 51] → 6, Knot[10, 52] → 6, Knot[10, 53] → 6, Knot[10, 54] → 6, Knot[10, 55] → 6,
  Knot[10, 56] → 6, Knot[10, 57] → 6, Knot[10, 58] → 6, Knot[10, 59] → 6, Knot[10, 60] → 6,
  Knot[10, 61] → 6, Knot[10, 62] → 6, Knot[10, 63] → 6, Knot[10, 64] → 6, Knot[10, 65] → 6,
```

```
Knot[10, 66] → 6, Knot[10, 67] → 6, Knot[10, 68] → 6, Knot[10, 69] → 6, Knot[10, 70] → 6,
Knot[10, 71] → 6, Knot[10, 72] → 6, Knot[10, 73] → 6, Knot[10, 74] → 6, Knot[10, 75] → 6,
Knot[10, 76] → 6, Knot[10, 77] → 6, Knot[10, 78] → 6, Knot[10, 79] → 6, Knot[10, 80] → 6,
Knot[10, 81] → 6, Knot[10, 82] → 6, Knot[10, 83] → 6, Knot[10, 84] → 6, Knot[10, 85] → 6,
Knot[10, 86] → 6, Knot[10, 87] → 6, Knot[10, 88] → 6, Knot[10, 89] → 6,
Knot[10, 90] → 6, Knot[10, 91] → 6, Knot[10, 92] → 6, Knot[10, 93] → 6,
Knot[10, 94] → 6, Knot[10, 95] → 6, Knot[10, 96] → 6, Knot[10, 97] → 6,
Knot[10, 98] → 6, Knot[10, 99] → 6, Knot[10, 100] → 6, Knot[10, 101] → 6,
Knot[10, 102] → 6, Knot[10, 103] → 6, Knot[10, 104] → 6, Knot[10, 105] → 6,
Knot[10, 106] → 6, Knot[10, 107] → 6, Knot[10, 108] → 6, Knot[10, 109] → 6,
Knot[10, 110] → 6, Knot[10, 111] → 6, Knot[10, 112] → 6, Knot[10, 113] → 6,
Knot[10, 114] → 6, Knot[10, 115] → 6, Knot[10, 116] → 6, Knot[10, 117] → 6,
Knot[10, 118] → 6, Knot[10, 119] → 6, Knot[10, 120] → 6, Knot[10, 121] → 6,
Knot[10, 122] → 6, Knot[10, 123] → 6, Knot[10, 124] → 6, Knot[10, 125] → 6,
Knot[10, 126] → 6, Knot[10, 127] → 6, Knot[10, 128] → 6, Knot[10, 129] → 6,
Knot[10, 130] → 6, Knot[10, 131] → 6, Knot[10, 132] → 6, Knot[10, 133] → 6,
Knot[10, 134] → 6, Knot[10, 135] → 6, Knot[10, 136] → 6, Knot[10, 137] → 6,
Knot[10, 138] → 6, Knot[10, 139] → 6, Knot[10, 140] → 6, Knot[10, 141] → 6,
Knot[10, 142] → 6, Knot[10, 143] → 6, Knot[10, 144] → 6, Knot[10, 145] → 6,
Knot[10, 146] → 6, Knot[10, 147] → 6, Knot[10, 148] → 6, Knot[10, 149] → 6,
Knot[10, 150] → 6, Knot[10, 151] → 6, Knot[10, 152] → 6, Knot[10, 153] → 6,
Knot[10, 154] → 6, Knot[10, 155] → 6, Knot[10, 156] → 6, Knot[10, 157] → 6,
Knot[10, 158] → 6, Knot[10, 159] → 6, Knot[10, 160] → 6, Knot[10, 161] → 6,
Knot[10, 162] → 6, Knot[10, 163] → 6, Knot[10, 164] → 6, Knot[10, 165] → 6}}
```