Dhannya's Conflict free colouring - Implementation

December 2017

1 Data Structure

- Interval -Class with left and right as attribute
- Point - Integer points between 1 to N.
- Sort - Using comparator

2 Introduction

These are the subroutine of the main algorithm.

- IsProper(setOfIntervals I)
- colourProper(setOfIntervals I)
- partitionInto(Intervals I,)
- Greedy CFC - Main algorithm

```
   Data: Intervals I
   Result: NO, if ∃i ∈ I which is fully contained in some other interval
   YES, otherwise
1  for i ∈ I do
2  |  for j ∈ I do
3  |  |  if i ≠ j and i ⊂ j then
4  |  |  |  return NO;
5  |  |  end
6  |  end
7  end
8  return YES;
```

Algorithm 1: isProperInterval

Data: Set of Intervals I, Points N
Result: Subset of points $S \subseteq N$ that can be uniquely colored

```
1  // I is pass by value ;
2  S = ∅ ;
3  Sort(I) ;
4  while I.getRightMost() ≠ NULL do
       /* Let p be the right most point in I                    */
5      I.removeAllIntervalsAt(p) ;
6      S.add(p) ;
7  end
8  return S;
```

Algorithm 2: colourProper

Data: Set of Intervals I, set of Points N
Result: Set of subsets $S \subseteq 2^S$ that can be uniquely colored

```
1  if isProperInterval(I) then
2      // Only one non-zero color is needed ;
3      return colourProper(I, N);
4  end
   /* Sort using the right most end point of the intervals; In
      case of ties sort with left end point                     */
5  Sort(I) ;
   /* Let the sorted interval be I[1], I[2], ..., I[m]         */
   /* Let B be the buckets or partitions; There are at most m/2
      buckets; Initially the buckets are empty                  */
6  B[1].add(I_1) ;
7  B[1].add(I_2) ;
8  for I_j ∈ I_3, ..., I_m do
9      i = 1 ;
10     while Partition(B[i], I_j) == NO do
11         i++
12     end
       /* I_j must go into some partition in B                  */
13 end
14 // The #non-zero bucket is equal to the #non-zero colours needed ;
15 i = 1 ;
16 S = ∅ ;
17 for B[i] ≠ EMPTY do
18     S = S∪ colourProper(B[i], N);
19 end
20 return S;
```

Algorithm 3: Greedy CFC

Data: Set of Intervals P in a partition B , New Interval I_i
Result: YES, if it is okay to add i to P, No otherwise.

```
1  // P is pass by reference. ;
2  if PSet.size() ≤ 2 then
3  |  P.add(I_i) ;
4  |  return YES ;
5  Q = P ∪ {I_i};
6  //Flag setting ;
7  // Let L be flag array initialized to 1 ;
8  for I_j ∈ Q do
9  |  // i represents the points by the interval I_i ;
10 |  if I_i ≠ I_j and I_j ⊂ I_i then  //I_j is contained in I_i
11 |  |  for k ∈ j \ i do
12 |  |  |  L[k] = 0 ;
13 |  |  end
14 |  else if I_i ≠ I_j and I_i ⊂ I_j then  //I_i is contained in I_j
15 |  |  for k ∈ i \ k do
16 |  |  |  L[k] = 0 ;
17 |  |  end
18 |  end
19 end
20 if FindAllZeroInterval(Q, L) then
21 |  return NO;
22 end
23 // Chopping ;
24 // TODA Let new intervals I'
25 if isProperInterval(I') == YES then
26 |  return YES ;
27 Else // call greedy algorithm with new interval. ;
```

Algorithm 4: PartitionInto