IIT Bhubaneswar

Start

RECOLORING INTERVAL GRAPHS WITH LIMITED RECOURSE BUDGET

Subject : ALGORITHMS

Submit by : AMAN DANGI

- Resource bound / resource budget - The number of allowed recolorings per step.
- It is NP-hard to approximate the chromatic number of an n-vertex graph.
- For interval graphs, linear time greedy algorithm achieves the optimum coloring.
- For online coloring, we know that each resulting interval graph will be k colorable, and k is known a priori.
- Maintaining a ck-coloring is referred to as c-approximation.

- We give an algorithm that maintains a 2k-coloring with an amortized recourse budget of $O(\log n)$.
- For maintaining a k-coloring with $k \le n$, we give an amortized upper bound of $O(k \cdot k! \cdot \sqrt{n})$.
- For unit interval graphs we give an algorithm that maintains a $(k + 1)$-coloring with at most $O(k^2)$ recolorings per step.
- For unit interval graphs we also give a lower bound of $\Omega(\log n)$ on the amortized recourse budget needed to maintain a k-coloring.
- If we does not maintain the exact colorings, then we can K-color in $O(k^2 ((\log n)^3))$ amortized time per update and querying for the color of a particular interval in $O(\log n)$ time.

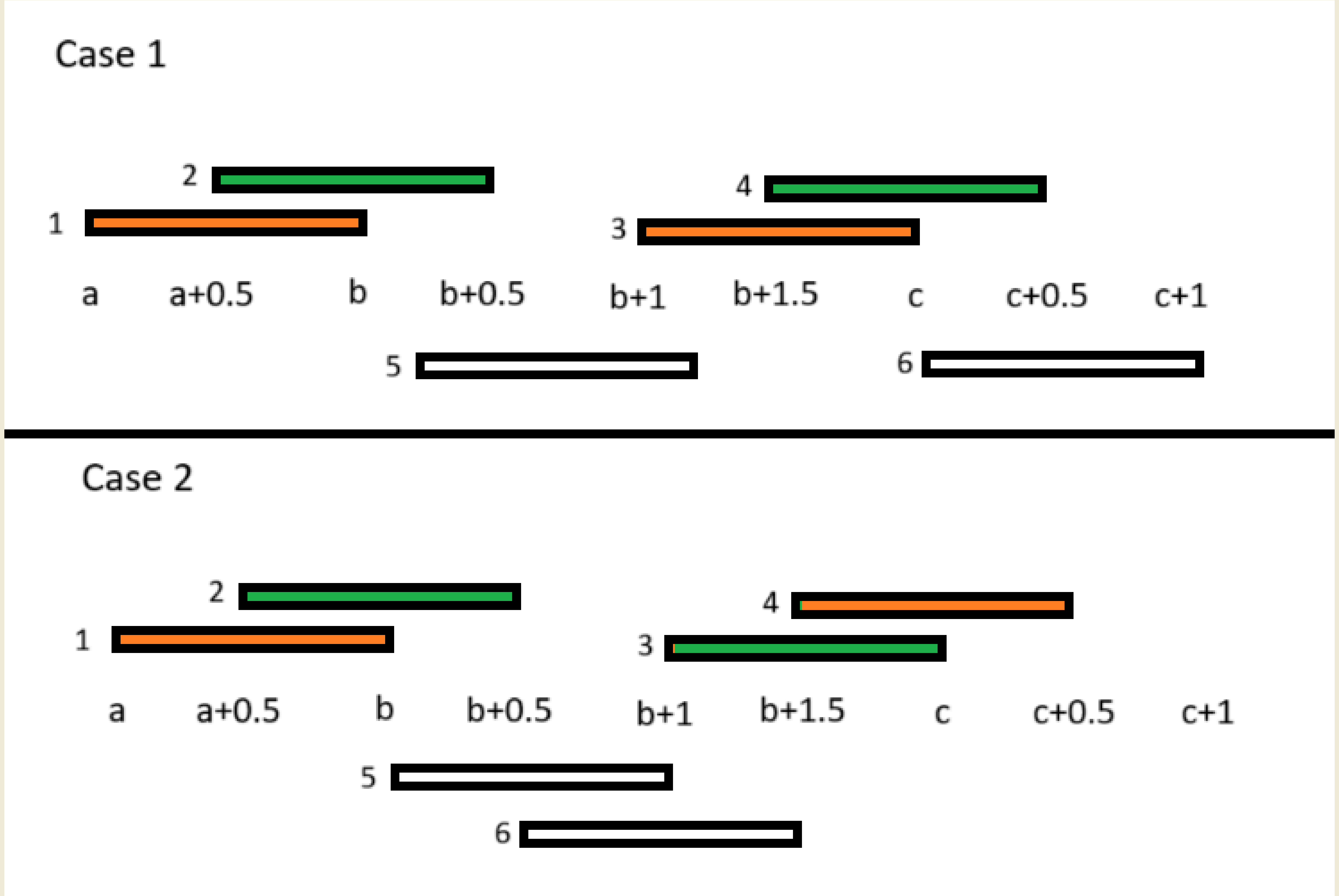
Theorem 1: Maintaining an optimum coloring of a 2-colorable unit interval graph requires an amortized recourse budget of Ω(log n).

We try to make G(i) from 2 G(i-1) and adding 2 more intervals in that.

First, let us understand the structure of G(0) -> it will have 2 intervals, [0,1) and [0.5,1.5), so it is trivial to note that it is 2 colorable.

Now since G(i-1) will be 2 colorable we can assume all the intervals of G(i-1) will be of 2 types - [a,b) and [a + 0.5,b + 0.5), and all the intervals of type [a,b) will be colored from 1 color and all the intervals of form [a + 0.5,b + 0.5) will be colored with another color, thus it is 2 colorable.

Now we want to construct G(i) from 2 G(i-1) and by adding 2 more intervals.



Case 1:
I can only color 6 as orange as it is overlapping with 4th interval, so I will have to color 5 as green, but it is overlaps with 2nd interval, so I will need to recolor either (1 and 2) or (3 and 4), so number of recolorings is equal to number of elements in G(i-1).

Case 2:
Same logic, I can color 5 as orange only, but 6 cannot be color green as it overlaps with 3 so recolor one of G(i-1), so here also recolorings equals number of intervals in G(i-1).

N(i) = 2*N(i-1) + 2, because for creating I am using 2 G(i-1) and adding 2 more intervals.

R(i) = 2*R(i-1) + N(i-1), because total recoloring for making G(i) will be recoloring of one of G(i-1) so N(i-1) is coming from there, and to create 2 G(i-1) I will need total 2*R(i-1) recoloring.

The base cases, N(0) = 2 as 2 intervals in G(0), and R(0) = 0 as no recoloring require to make G(0).

Solving for N(i) gives, N(i) = 2^(i+2) - 2
and R(i) gives R(i) = (i-1)*2^(i+1) + 2

And now by recoloring i intervals I am creating a new graph of 2*i+2 nodes, so for creating a graph of 'n' nodes I will require, n/2+n/4+... ---->>> doubt

$$\frac{R(i)}{N(i)} = \frac{(i-1)}{2(1-2^{-i-2})} + \frac{1}{2^{i+1}-1}$$

Theorem 2: There exists an algorithm which maintains a (k + 1)-coloring of a k-colorable unit interval graph with $O(k^2)$ worst case recourse budget per update.

We partition the current instance I into smaller instances $I_1, I_2, \ldots, I_m$ and separators between them.

Each instance is of size at least lk(except for the last one) and at most 2lk + k for $l = \max\{4, k + 1\}$.

Construction of Si -> if one interval grows above 2lk+k, then we pick lk intervals in one I(new) and then we choose value of p to be r of the last interval [l,r) of I(new) and at max k intervals will intersect it as k is chromatic number thus in next I(new) also at least lk elements will be there, thus preserving our construction property that it's size should be at least lk.

So, $I = I_1 \cup S_1 \cup I_2 \cup S_2 \ldots \cup S_{m-1} \cup I_m$, where $m \in \Theta(n/(k^2))$

Lemma 3: Let I be a k-colorable unit interval instance. If |I| ≥ lk for l = max{4, k + 1}, then, for any fixed coloring on ξl(I) and ξr(I) using colors from [k], one can complete this coloring on I using colors from [k + 1].

We draw the intervals of I as arcs on the north half of a circle, in a way that preserves the intersection relation. Let p be the south pole of the circle, i.e., the point extending the most to the south. For each pair of intervals (I1, I2) ∈ ξl(I) × ξr(I) such that I1 and I2 are precolored with the same color, we stretch I1 (respectively I2) anticlockwise (respectively clockwise) so that they reach p and then glue them together to form the same arc. The remaining intervals of ξl(I) and ξr(I) are only stretched to reach (and intersect) p and are not glued with anything.

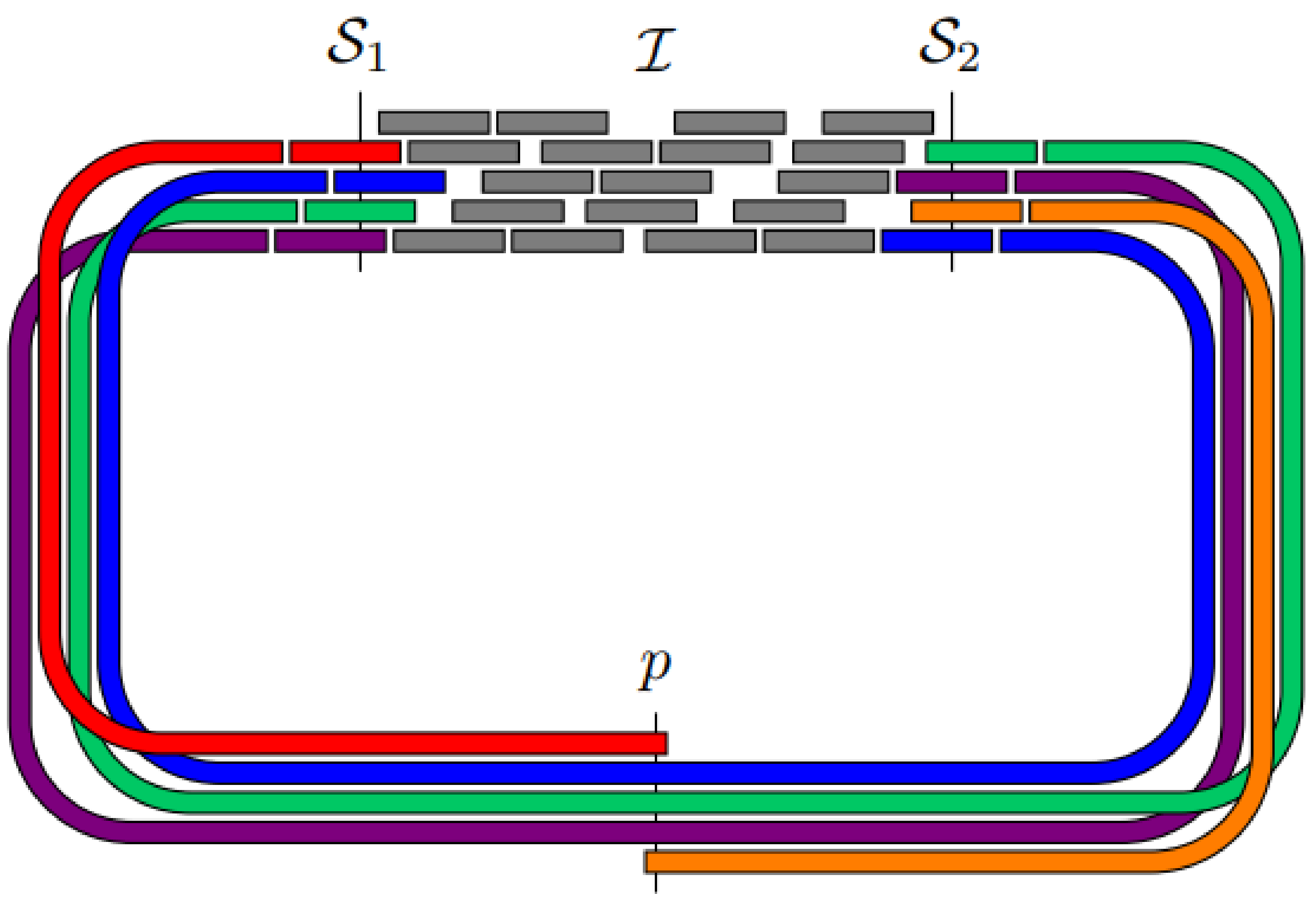


Illustration of the proof of Lemma 3 for $k = 5$.

Lemma 4: Let G be a circular arc graph, L(G) be the maximum number of arcs intersecting a common point on the circle, and l(G) be the smallest number of intervals that cover the circle. If l(G) ≥ 5 then ceil { [(l(G)-1)/(l(G)-2)]*L(G) } colors suffice to color G and there is a linear time coloring algorithm.

Since chromatic number is k, that implies L(G) <= k.

Since |I| >= lk and and it is unit interval graph, so smallest number of intervals to cover circle will be at least lk+1 or we can also l+1 and l is at least 4, so l(G) >= 5, so I can apply the lemma.

So { [(l(G)-1)/(l(G)-2)]*L(G) } is <= k+1, so we can say we can color it in linear time with k+1 colors.

Have to see paper 34 for implementation.

When a new interval Inew is added, it either fits into an instance Ii or it belongs to a separator Sj . In the first case, we recolor Ii ∪ {Inew} consistently with the current coloring on Si−1 and Si . In the second case, we color the new interval Inew with the first color not used on Sj and recolor Ij and Ij+1 consistently with the current coloring on Sj−1, Sj , and Sj+1.

Theorem 5: There is an algorithm maintaining a 2-approximate coloring of an interval graph with amortized recourse budget $O(\log n)$.

Lemma 6: There is an online algorithm which receives an interval graph G in an online way and produces a partition of G into subgraphs $P_1, \ldots, P_\omega$, where each P_i is a sum of disconnected paths and ω is a clique number of G.

For this I will have to read research paper number 20 properly

Lemma 7: There is an incremental algorithm which uses 2 colors on a sum of disconnected paths P with $n \log_2 n$ total changes, where n is a size of P.

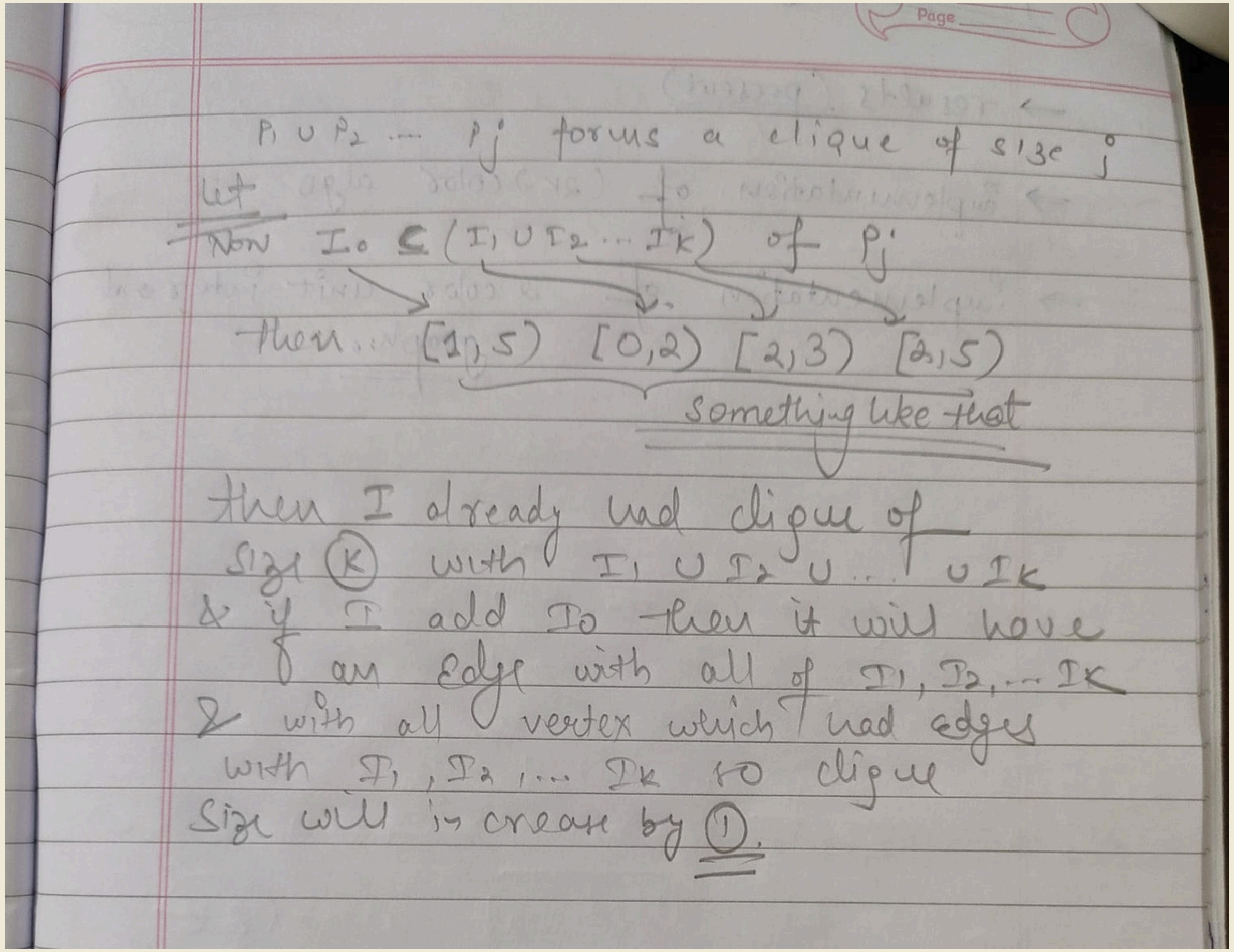
For this I will try to maintain 2 properties:

1. For any $j \le \omega$ each clique in $P1 \cup P2 \cup . . . \cup Pj$, has size at most j.
2. For any $j \le \omega$ and for any vertex $u \in Pj$ there is a clique in $P1 \cup P2 \cup . . . \cup Pj-1 \cup \{u\}$ of size j.

While insuring these properties I will end up with these:

Claim 8. There is no interval in Ij which is covered by the rest of the intervals from Ij.

Because if it is there then already I had P1 union P2 union …… P(w) whose clique size was w, but if one more interval is there which is covered already then it will increase the size of clique further by 1, which will result in contradiction of our 2 invariants.



Claim 9: Each vertex in Pj has at most two neighbours in Pj.

This follows from the fact that no interval should be covered completely by other intervals, let say v0 has 3 neighbours v1, v2, v3 and let I0, I1, I2, I3 be the corresponding intervals, then let say union of all gives [l, r) and at least one interval will be there which won't have any of l, or r. So we can say that belongs in the other intervals, which contradicts our claim 8, hence it can have at max 2 neighbours.

Now proof of lemma 7: When new vertex v is coming, it combines two paths. If neighbours of v have the same color then the algorithm colors vertex v on the other one. If neighbours of v have the different colors then the algorithm recolors the shortest path. The given vertex u was recolored when the length of the path containing u increased by at least twice. This causes the vertex u to be recolored at most log2 n times. Which gives the total number of recoloring equal n log2 n.

interval graphs are also chordal, so they can not contain simple cycles

- It is a 3 competitive algorithm i.e. the number of colors used are thrice the chromatic number.
- Color assigned will be a tuple: (p(v), o(v)), where p(v) is level and o(v) is the offset.
- Aim is how to find p(v) and o(v)

Calculating $p(v)$

adding v_i

→ create induced subgraph $G_r(v_i)$

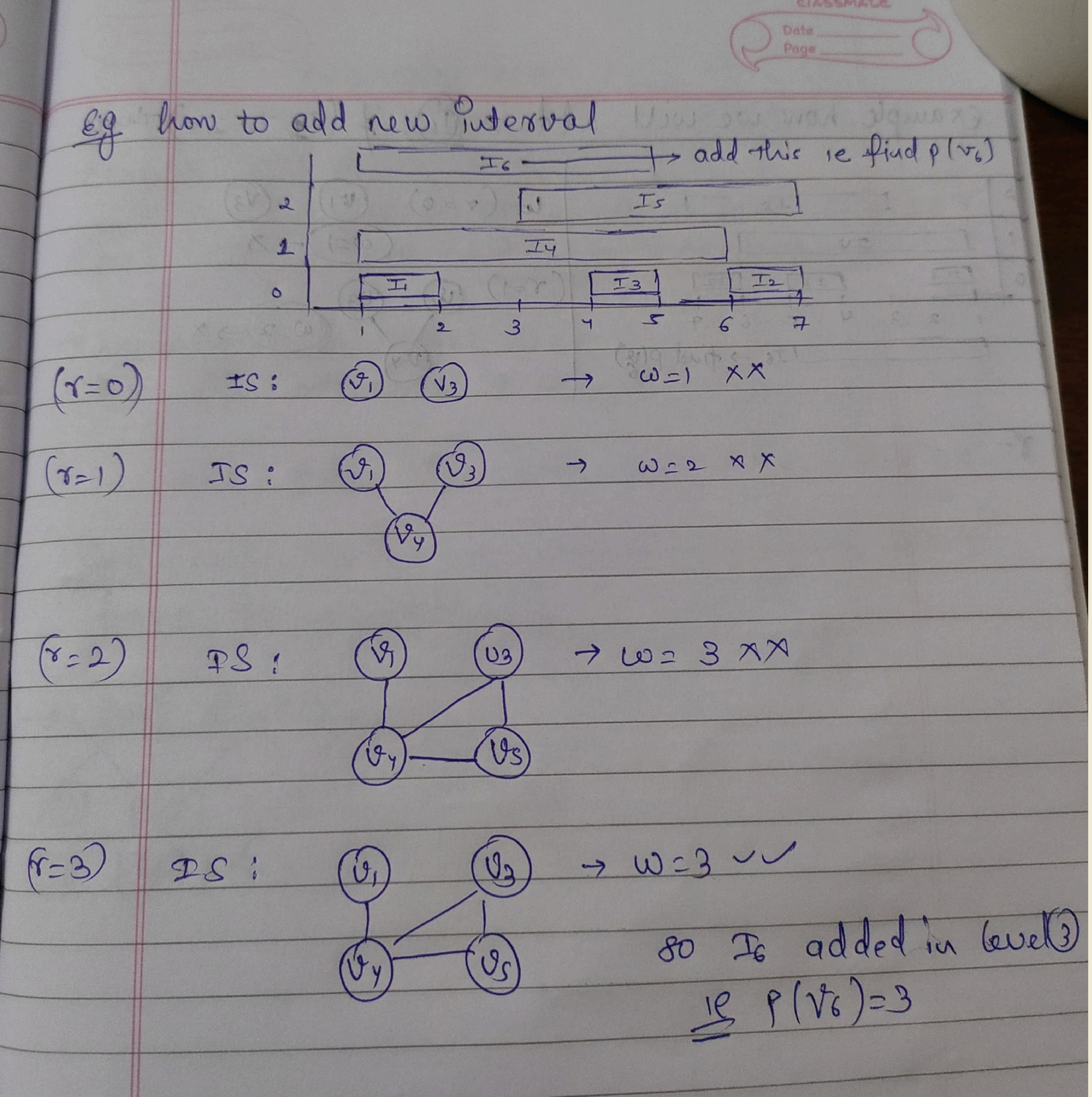
↳ have to consider all I which have $p(v_j) \le r$

and $p(v_i)$ will be $\min\{r \mid \omega(G_r(v_i)) \le r\}$

→ basically I will start from $r=0$, if it holds then fine otherwise ↑ r gradually.

and those v_j should have edge with v_i

and while constructing induced sub G we don't have to consider v_i



Properties of p(v):

- Number of levels will be less than equal to chromatic number.
- **On each level, each Interval will have at max 2 neighbours.**

Computing o(v):

- On each levels each interval will have at max 2 neighbours, so color them from {1, 2, 3}.