Vertex Coloring and Upper Bounds

Dr. Rajiv Misra
Associate Professor
Dept. of Computer Science & Engg.
Indian Institute of Technology Patna
rajivm@iitp.ac.in

Preface

Recap of Previous Lecture:

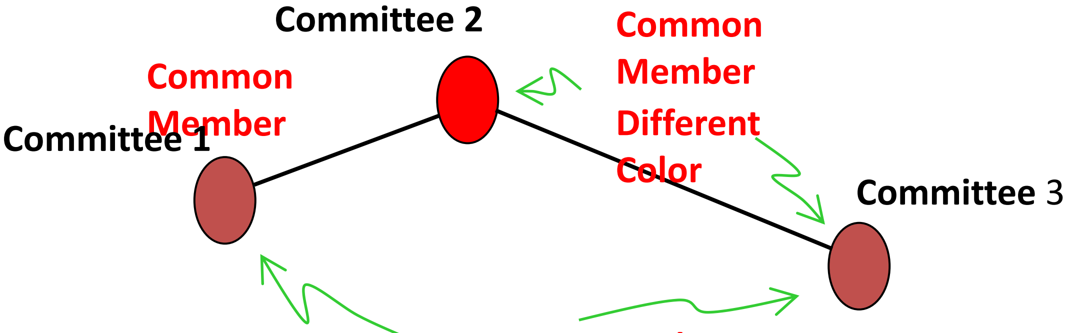
In previous lecture, we have discussed the Network Flow Problems *i.e.* Maximum Network Flow, *f*-augmenting path, Ford-Fulkerson labeling algorithm and Max-Flow Min-cut Theorem.

Content of this Lecture:

In this lecture, we will discuss Graph Coloring *i.e.* Vertex Coloring and Upper Bounds.

Vertex Coloring and Upper Bounds

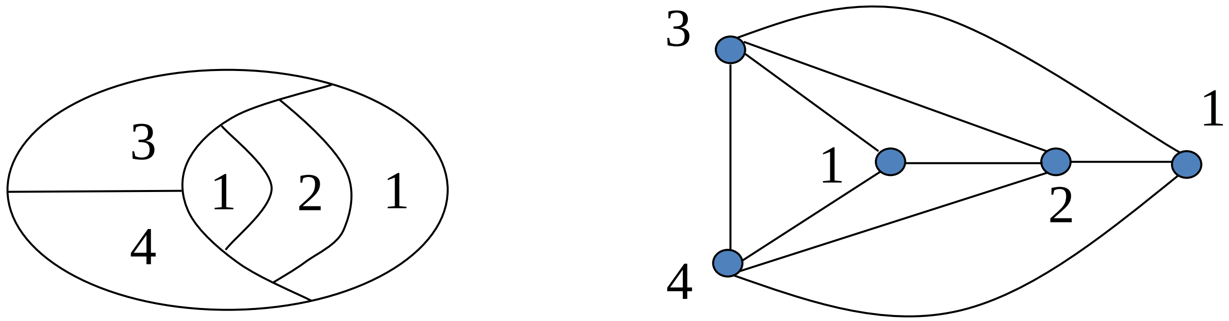
- The committee-scheduling example uses graph coloring to model avoidance of conflicts.
- Similarly, in a university we want to assign time slots for final examinations so that two courses with a common student have different slots. The number of slots needed is the chromatic number of the graph in which two courses are adjacent if they have a common student.



No Common Member
Same Color OK
Same time slot OK

Map Region Coloring

- Coloring the regions of a map with different colors on regions with common boundaries is another example.
- The map on the left below has five regions, and four colors suffice. The graph on the right models the “common boundary” relation and the corresponding coloring. Labeling of vertices is our context for coloring problems.



Vertex coloring 5.1.1

- A ***k*-coloring** of a graph G is a labeling $f: V(G) \to S$, where $|S| = k$ (often we use $S = [k]$). The labels are **colors**; the vertices of one color form a **color class**.
- A k-coloring is **proper** if adjacent vertices have different labels.
- A graph is ***k*-colorable** if it has a proper k-coloring. The **chromatic number** $\chi(G)$ is the least k such that G is k-colorable.

Remark 5.1.2

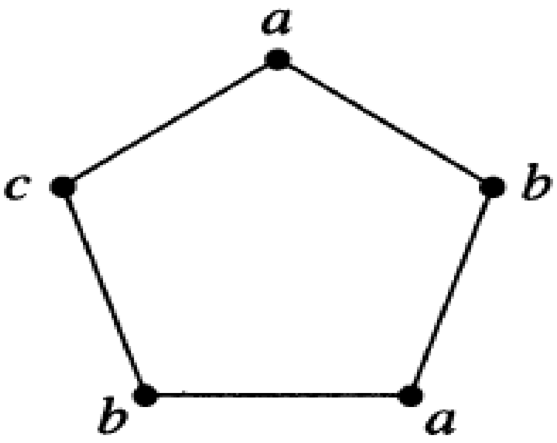
- In a proper coloring, each color class is an independent set, so G is k-colorable if and only if V(G) is the union of k independent sets.

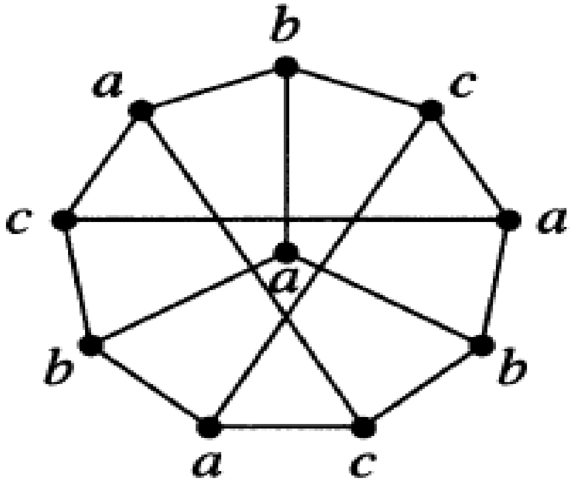
- Thus “k-colorable” and “k-partite” have the same meaning. (The usage of the two terms is slightly different. Often “k-partite” is a structural hypothesis, while “k-colorable” is the result of an optimization problem.)

Example 2-Colorable and 3-colorable

5.1.3

- Since a graph is 2-colorable if and only if it is bipartite, C_5 and
- The Peterson graph have chromatic number at least 3. Since they are 3-colorable, as shown below, they have chromatic number exactly 3.





k-chromatic 5.1.4

- A graph *G* is ***k*-chromatic** if $\chi(G) = k$.
- A proper *k*-coloring of a *k*-chromatic graph is an **optimal coloring.**

k-critical graphs for small k 5.1.5

If $\chi(H) < \chi(G) = k$ for every **proper** subgraph H of G, then G is **color-critical** or ***k*-critical.**

- Properly coloring a graph needs at least two colors if and only if the graph has an edge. Thus K_2 is the only 2-critical graph (similarly, K_1 is the only 1-critical graph). Since 2-colorable is the same as bipartite, the characterization of bipartite graphs implies that the 3-critical graphs are the odd cycles.
- We can test 2-colorability of a graph G by computing distances from a vertex x (in each component). Let $X = \{u \in V(G): d(u,x) \text{ is even}\}$ and let $Y = \{u \in V(G): d(u,x) \text{ is odd}\}$. The graph G is bipartite if and only if X,Y is a bipartition, meaning that G[X] and G[Y] are independent sets.
- No good characterization of 4-criticial graphs or test for 3-colorability is known.

Clique number 5.1.6

- The ***clique number*** of a graph *G*, written ω(*G*), is the maximum size of a set of pairwise adjacent vertices (clique) in *G*.

- α(G) is used for the independence number of G; the usage of ω(*G*) is analogous.

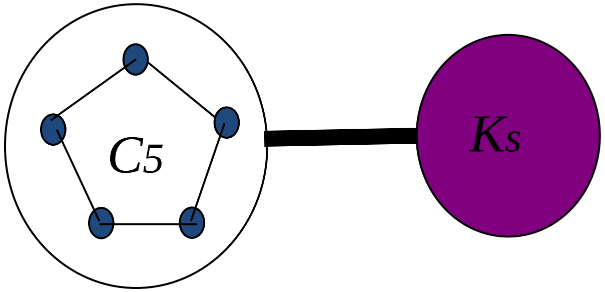
Proposition 5.1.7 For every graph G, $\chi(G) \geq \omega(G)$ and $\chi(G) \geq n(G)/\alpha(G)$.

- **Proof:** The first bound holds because vertices of a clique require distinct colors. The second bound holds because each color class is an independent set and thus has at most $\alpha(G)$ vertices.

  - $\chi(G)$: chromatic number
  - $\omega(G)$: clique number
  - $\alpha(G)$: independence number

Example 5.1.8.

- $\chi(G)$ *may exceed* $\omega(G)$.
- For $r \geq 2$, let $G = C_{2r+1} \vee K_s$ (the join of C_{2r+1} and K_s) Since C_{2r+1} has no triangle, $\omega(G) = s+2$.
- Properly coloring the induced cycle requires at least three colors. The *s*-clique needs *s* colors. Since every vertex of the induced cycle is adjacent to every vertex of the clique, these *s* colors must differ from the first three, and $\chi(G) \geq s+3$. We conclude that $\chi(G) > \omega(G)$.

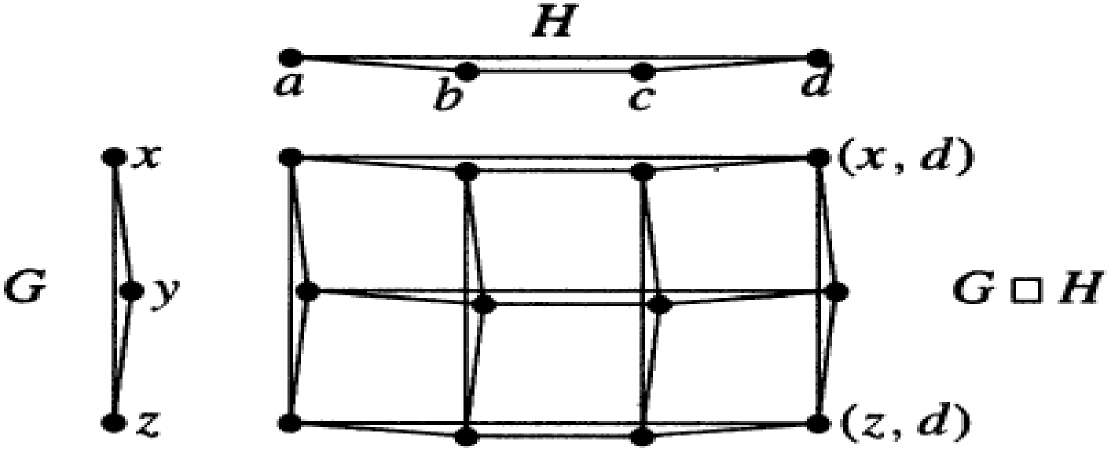


Definition: Cartesian Product 5.1.9

- The **cartesian product** of G and H, written G □ H, is the graph with vertex set V(G) x V(H) specified by putting (u,v) adjacent to (u',v') if and only if (1) u=u' and vv'ϵ E(H), or (2) v = v' and uu' ϵ E(G).

Example 5.1.10

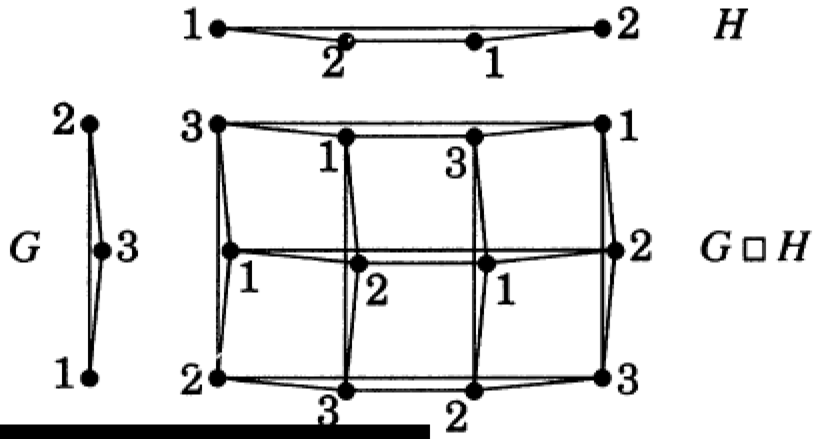
- The cartesian product operation is symmetric; $G \square H \cong H \square G$. Below we show $C_3 \square C_4$ .The hypercube is another familiar example: $Q_k = Q_{k-1} \square K_2$ when $k \geq 1$. The m-by-n **grid** is the cartesian product $P_m \square P_n$.
- In general, $G \square H$ decomposes into copies of H for each vertex of G and copies of G for each vertex of H. We use $\square$ instead of x to avoid confusion with other products, reserving x for the cartesian product of vertex sets. The symbol $\square$, introduced by Nešetřil, evokes the identity $K_2 \square K_2 = C_4$.



Proposition : (Vizing [1963], Alberth [1964]) 5.1.11

$\chi(G \,\square\, H) = \max\{\chi(G), \chi(H)\}$

- **Proof:** The cartesian product of G □ H contains copies of G and H as subgraphs, so $\chi(G \,\square\, H) \geq \max\{\chi(G), \chi(H)\}$
- Let $k=\max\{\chi(G), \chi(H)\}$. To prove the upper bound, we produce a proper k-coloring of G □ H using optimal colorings of G and H. Let g be a proper $\chi(G)$-coloring of G, and let h be a proper $\chi(H)$-coloring of H. Define a coloring *f* of G □ H by letting *f*(u,v) be the congruence class of g(u)+h(v) modulo k. Thus *f* assigns colors to V(G □ H) from a set of size k.
- We claim that *f* properly colors G □ H . If (u,v) and (u',v') are adjacent in G □ H , then g(u) + h(v) and g(u') + h(v') agree in one summand and differ by between 1 and k in the other. Since the difference of the two sums is between 1 and k, they lie in different congruence classes modulo k.



Upper Bounds

- Most upper bounds on the chromatic number come from algorithms that produce colorings.
  - For example, assigning distinct colors to the vertices yields $\chi(G) \leq n(G)$.
  - This bound is best possible, since $\chi(K_n) = n$, but it holds with equality only for complete graphs.

We can improve a “best-possible” bound by obtaining another bound that is always at least as good.

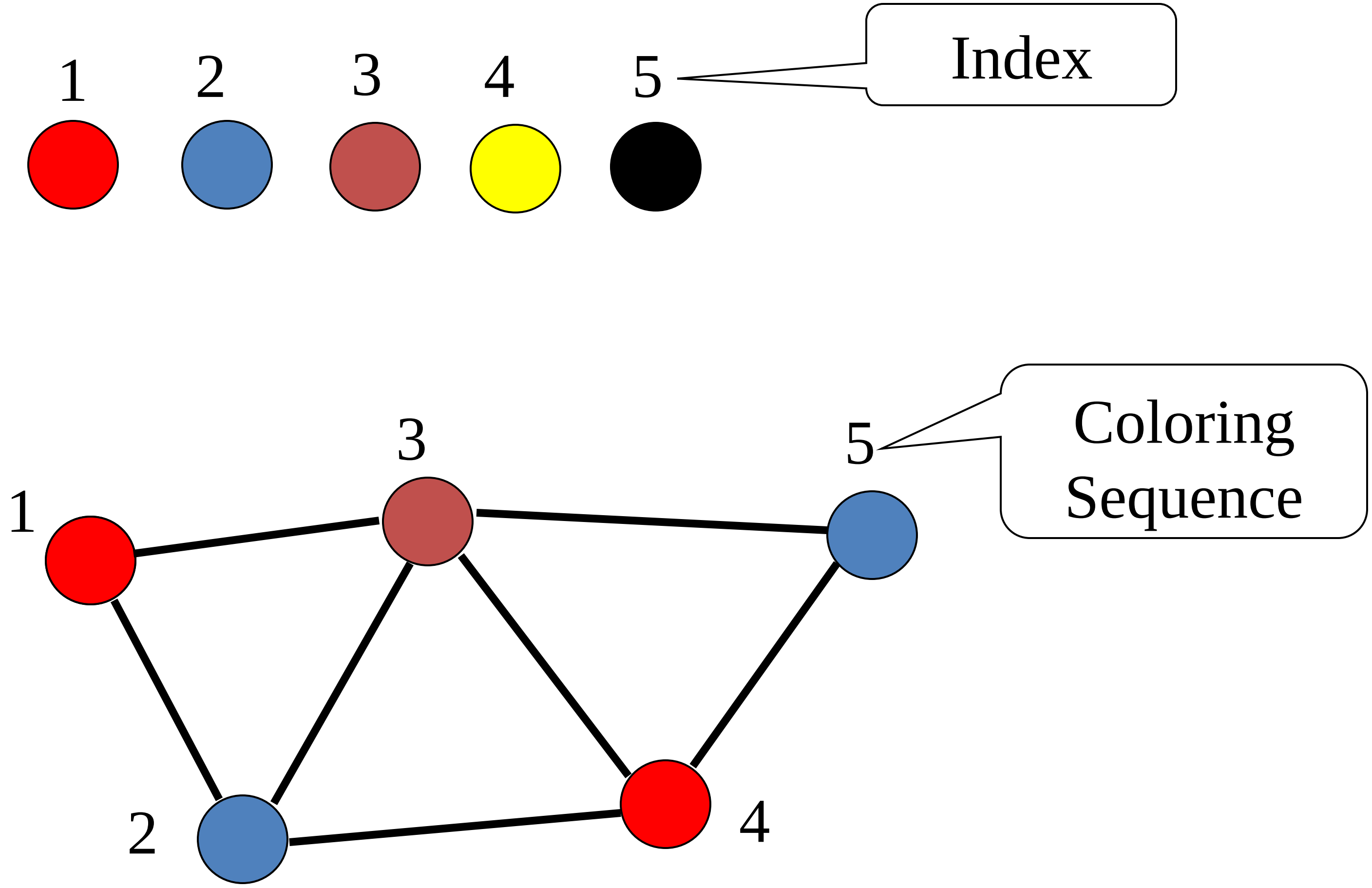
For example $\chi(G) \leq n(G)$ uses nothing about the structure of G;

we can do better by coloring the vertices in some order and always using the “least available ” color.

Algorithm: Greedy Coloring 5.1.12

- The **greedy coloring** relative to a vertex ordering $v_1, \ldots, v_n$ of $V(G)$ is obtained by coloring vertices in the order $v_1, \ldots\ldots, v_n$, assigning to v_i the smallest-indexed color not already used on its lower-indexed neighbors.

Example of Greedy Coloring Algorithm



Proposition: $\chi(G) \leq \Delta(G) + 1$ 5.1.13

Proof:

- In a vertex ordering, each vertex has at most $\Delta(G)$ earlier neighbors, so the greedy coloring cannot be forced to use more than $\Delta(G) + 1$ colors. This proves constructively that $\chi(G) \leq \Delta(G) + 1$.

- The bound $\Delta(G) + 1$ is the worst upper bound that greedy coloring could produce (although optimal for complete graphs and odd cycles). Choosing the vertex ordering carefully yields improvements. We can avoid the trouble caused by vertices of high degree by putting them at the beginning, where they won't have many earlier neighbors.

Proposition: (Welsh-Powell [1967]) 5.1.14

- If a graph G has degree sequence $d_1 \geq \ldots\ldots \geq d_n$, then $\chi(G) \leq 1 + \max_i \min\{d_i, i-1\}$

Proof:

- We apply greedy coloring to the vertices in nonincreasing order of degree. When we color the *i*th vertex v_i, it has at most $\min\{d_i, i-1\}$ earlier neighbors, so at most this many colors appear on its earlier neighbors. Hence the color we assign to v_i is at most $1 + \min\{d_i, i-1\}$. This holds for each vertex, so we maximize over *i* to obtain the upper bound on the maximum color used.

Remark

- The bound in Proposition 5.1.14 is always at most $1+\Delta(G)$, so this is always at least as good as Proposition 5.1.13. It gives the optimal upper bound in Example 5.1.8, while $1+\Delta(G)$ does not.

- In Proposition 5.1.14, we use greedy coloring with a well-chosen ordering. In fact, every graph G has some vertex ordering for which the greedy algorithm uses only $\chi(G)$ colors. Usually it is hard to find such an ordering.

- Our next example introduces a class of graphs where such an ordering is easy to find. The ordering produces a coloring that achieves equality in the bound $\chi(G) \geq \omega(G)$.

Example: Register allocation and interval graphs

- A computer program stores the values of its variables in memory. For arithmetic computations, the values must be entered in easily accessed locations called ***registers***.
- Registers are expensive, so we want to use them efficiently. If two variables are never used simultaneously, then we can allocate them to the same register.
- For each variable, we compute the first and last time when it is used. A variable is ***active*** during the interval between these times.

Example: Register allocation and interval graphs

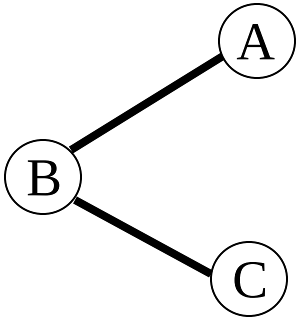
continue

Interval graph

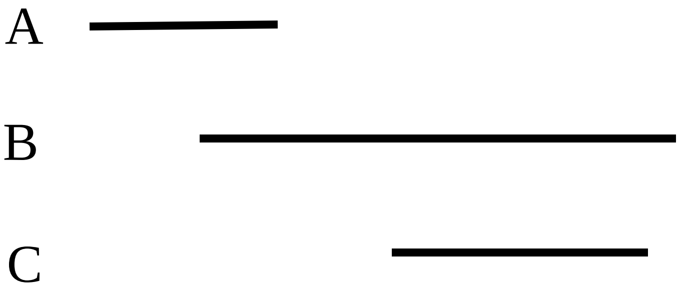
- We define a graph whose vertices are the variables.
- Two vertices are adjacent if they are active at a common time.
- The number of registers needed is the chromatic number of this graph.
- The time when a variable is active is an interval, so we obtain a special type of representation for the graph.

Interval Representation and interval graphs continue

- An interval representation of a graph is a family of intervals assigned to the vertices
  - so that vertices are adjacent if and only if the corresponding intervals intersect.
- A graph having such a representation is an interval graph.



Interval graph

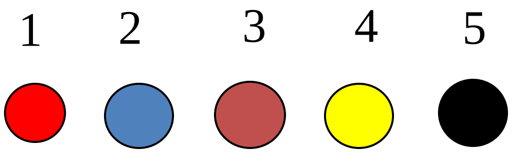


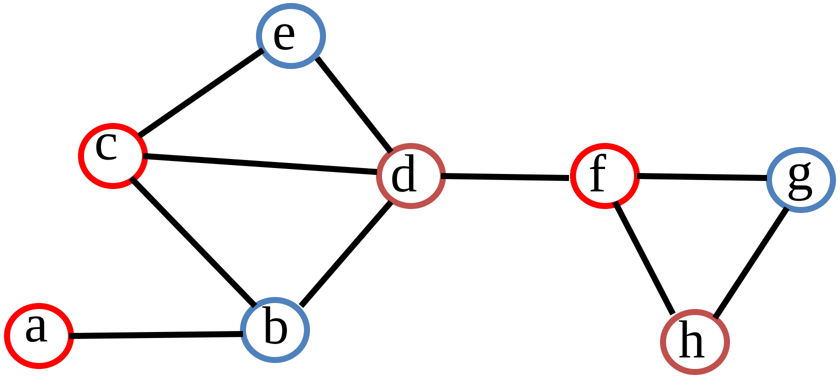
Interval representation

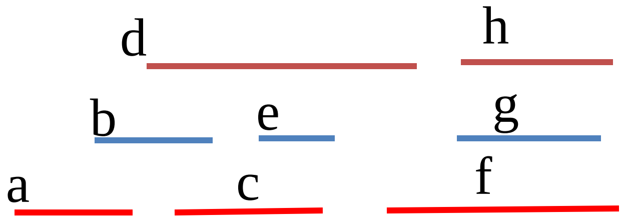
Example: Register allocation and interval graphs

continue

- For the vertex ordering *a, b, c, d, e, f, g, h* of the interval graph below, greedy coloring assigns 1, 2, 1, 3, 2, 1, 2, 3, respectively, which is optimal. Greedy colorings relative to orderings starting *a* , *d*, … use four colors.







Proposition 5.1.16. If G is an interval graph, then $\chi(G) = \omega(G)$

Proof:

- Order the vertices according to the left endpoints of the intervals in an interval representation.
- Apply greedy coloring, and suppose that x receives k, the maximum color assigned.
- Since x does not receive a smaller color, the left endpoint a of its interval belongs also to intervals that already have colors 1 through k-1.
- These intervals all share the point a, so we have a k-clique consisting of x and neighbors of x with colors 1 through k-1.
- Hence $\omega(G) \geq k \geq \chi(G)$. Since $\chi(G) \geq \omega(G)$ always, this coloring is optimal.

Remark 5.1.17

- The greedy coloring algorithm runs rapidly. It is “on-line” in the sense that it produces a proper coloring even if it sees only one new vertex at each step and must color it with no option to change earlier colors.

- For a random vertex ordering in a random graph, greedy coloring almost always uses only about twice as many colors as the minimum, although with a bad ordering it may use many colors on a tree.

Conclusion

- In this lecture, we have discussed k-coloring of a graph, optimal coloring, clique number, cartesian product, Upper bounds *i.e.* greedy coloring, register allocation and interval graphs.

- In upcoming lecture, we will discuss the Brooks' Theorem.