CSE525 Lec22
Approximation algo

•••

Debajyoti Bera (M21)

Non-decision problems

For NP-completeness, need decision problems.

Problems that are not decision problems can be ...

- Function problems (Find a colouring of a graph using at most 3 colours)
- Counting problems (Count the number of 3-colourings of a graph)
- Optimization problems (Optimize the number of colours needed to colour a graph)

How to deal with optimisation versions of NP-complete decision problems?

→ algorithms are probabilistic, $A(x) \to$ random, correct w/p $\geq 99\%$

randomised algorithms

→ algorithms return slightly inaccurate results } approximation algorithms

Approximation Ratio

APPROX = solution of approximate algorithm, OPT = optimal solution

r-approx algo.

Approx. algo is r-relative if …

2-approx. algorithm for Knapsack

$APPROX \in [\frac{OPT}{2}, \dots, OPT]$

- Maximization problem: $\frac{OPT}{r} \leq APPROX \leq OPT \equiv APPROX \leq OPT \leq r \cdot APPROX$
- Minimization problem: $OPT \leq APPROX \leq r \cdot OPT \equiv \frac{APPROX}{2} \leq OPT \leq APPROX$

Approx. algo ~~has~~ is r-absolute if …

- Maximization problem: $OPT - r \leq APPROX \leq OPT$
- Minimization problem: $OPT \leq APPROX \leq OPT + r$

$ApproxVC \leq 2 * OPT$

$\rightarrow Approx \leq f(\#edges)$

$\rightarrow OPT \geq f(\#edges)$

Given G, find a vertex cover with the smallest size.

Minimum Vertex Cover

$OPTVC \leq ApproxVC \leq 2 * OPTVC$

$\rightarrow$ 2 (rel) approx.

E' = set of edges chosen in (1)

Edges in E' do not share a vertex

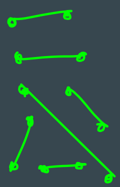
$\boxed{\text{Size of any VC} \geq |E'|}$

1. Pick an edge (u,v)
2. Add one of u or v to VC [based on some criteria]
3. Remove adjacent edges from added node
4. Goto 1.

O(E.V)

1. Pick an edge (u,v)
2. Add both u & v to VC
3. Remove all adjacent edges from u & v
4. Goto 1.

O(EV)



Q: Consider 2nd Algo and denote the size of its output as ApproxVC.. Show that in OptVC (optimum vertex cover) at least ApproxVC/2 vertices must be present.

Algorithmic obs. $ApproxVC = 2 * |E'|$

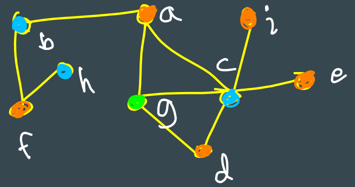
Q: Does ApproxVC return a VC? What is its (relative) approximation ratio?

Trivial obs. $\rightarrow OPTVC \geq |E'|$

Graph theoretic $\Rightarrow ApproxVC \leq 2 * OPTVC$

Fact: VC can be approximated but not to a high quality.

- Dinur Safra 2002: VC cannot be approximated to any constant ~~<~~= 1.36
- Khot Regev 2003: VC cannot* be approximated to any constant ~~<~~= 2 !!!



Chromatic number (CHR)

greedy

Chromatic number of G = optimal number of colours

Polynomial-time algorithm is unlikely.

What about polynomial-time algorithms returning “ad-hoc” solutions?

Col 1   Col 2 . . .   Col n

Greedy Algorithm (G): Colour greedily all vertices in some order. Number of colours?

v_1 : use Col 1 . for every vertex $v_2, v_3 \dots v_n$, colour v_i using the smallest possible colour

Trivial Algorithm (G): Colour every vertex using different colour. Number of colours?

$|V|$

Which algorithm is better ? How to evaluate quality ?

Greedy Algorithm for CHR

Claim: col(v_i) $\leq$ col k

$v_1 \quad v_2 \; \ldots \quad v_k$

$c_1 \quad c_2 \; \ldots \quad c_k$

atmost $c_1 \cdots c_{k-1}$ have been used.
$\therefore$ c_k is a valid colour for v_k

Claim: col(v_i) $\leq$ col(deg(v_i) + 1)

N = neighbours of v_i
$|N|$ = deg(v_i)
Atmost $|N|$ colours are used to colour N.
$\therefore$ (N+1) is a valid colour for v_i.

$c_1 \quad c_2 \quad c_3 \quad c_4 \quad c_5 \ldots$
atleast one will not be used by vertex in N.

Greedy(G):

Order vertices in any order: v1 v2 ...
Greedily assign colours from [c1, c2, ...] to every vi in that order (ensure no conflict)

What can be said about output of Greedy(G)?

Q: Show that colour of vi <= min{ deg(vi)+1, i }

Q: Give an upper bound on the number of colours?

n vertex, Δ: max degree $\quad \Delta + 1$

Q: Show a graph and ordering for which this performs badly!

G

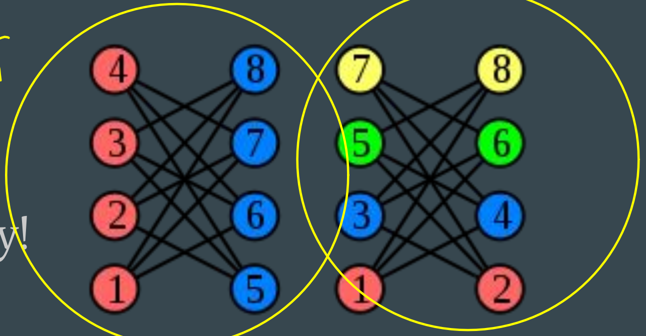


Image: wikipedia.com

Ex. Extend this example to any constant r.

Q: Form a strategy (ordering) to colour using few colours.

100 50 17 8 6 4 4 3 3 3 2 2 1 1 1 1 1 $\quad$ CHR(G) $\leq$ 2
Greedy(G) = 4

$\therefore$ Greedy cannot be r-relative approx. for any $r < 2$.

1-absolute CHR is not easy

if $CHR(G) \le k$: $k \le CHRAlgo(G) \le k+1$

Cannot decide

if $CHR(G) \geq k+1$: $k+1 \le CHRAlgo(G)$

def IsCHR(G,k):
Create H in the manner below // $CHR(H) = 2 * CHR(G)$
$r \leftarrow CHRAlgo(H)$
// $r = CHR(H)$ or $CHR(H)+1$
$= 2*CHR(G)$ or $2*CHR(G)+1$

Polytime CHRAlgo(G) : Output either CHR(G) or CHR(G)+1

Q: Reduce IsCHR to CHRAlgo IsCHR(G,k) = "Is chrom. num. <= k?"

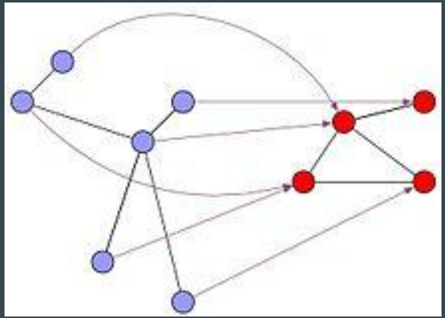
NP complete

IsCHR(G,k): Decide which case is true?

(a) CHR(G) <= k OR CHR(G) >= k+1 (b)

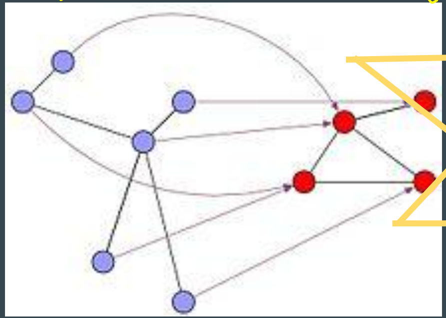
$CHR(H) \le 2k$ $CHR(H) \geq 2k+2$ $r \geq 2k+2$

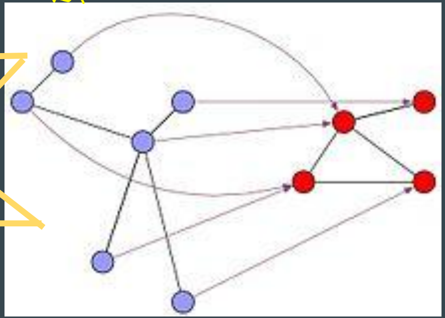
G



$r \le CHR(H)+1$
$r \le 2k+1$

H





if $r \le 2k+1$ output yes
else: $r \geq 2k+2$ output no.

CHR(G) <=k **OR** CHR(G) >= k+1 **IFF** CHR(G') <= ??? **OR** CHR(G') >= ???

Extend: r-absolute appr. is not possible for any constant r.

IFF CHRAlgo(G') <= ?? **OR** CHRAlgo(G') >= ??