Lecture 16  Approximation Algorithms continued.

- Graph coloring
- Max Cut - 2ways
- Metric TSP
Graph Coloring

How many colors needed for coloring nodes of $G$ so that no two adjacent nodes get same color?

Decision problems:

Is $G$ 2-colorable?
Is $G$ 3-colorable?

A type of approximation algorithm:

Given $G$, known to be 3-colorable

Output 3-in-coloring of $G$

First, an easy fact:

Fact 1: Any graph $G$ of degree $d$ can be colored with $d+1$ colors in poly time.

Why?
Consider nodes in arbitrary order.
Poly time algorithm (since $d$ neighbors, there is always $\geq 1$ such color)
So our only problem is nodes of deg > d?

how can we get rid of these?

Fact 2. \( G \) is 3-colorable. Define \( N(v) = \{ w \mid \text{w is a nbr of } v \} \)

Then, for any \( v \), \( N(v) = \{ w \mid \text{w is a nbr of } v \} \)

+ edges \( E_N(v) = \{ (u, w) \mid (u, w) \in E \} \)

Then \( G' = (N(v), E_N(v)) \)

is bipartite.

Why? in 3-coloring, nodes in \( N(v) \)

must be assigned different color than \( v \)

+ only 2 colors left.

This coloring doesn't violate edges between them.
And now, an Algorithm:

Algorithm

While \exists v \text{ st. } \deg(v) \geq \sqrt{n} \ do

- color \( v \) with color "1"
- color \( N(v) \) with 2 new colors

bipartite

remove \( v \) \& \( N(v) \)

Color remaining graph with \( \sqrt{n} \) new colors

- Runtime polynomial? \( \checkmark \) since bipartite coloring \( \mathcal{E}P \)
  + "greedy" proof of fact 1 gives poly time algorithm

- Correctness? can reuse "1" in while loop, since remove node + all neighbors

- 3\( \sqrt{n} \) colors? "While" loop executed \( \leq \sqrt{n} \) times, since each time it reduces number of nodes by \( \sqrt{n} + 1 \)

\( \Rightarrow \) # colors used in while loop \( \leq 1 + 2\sqrt{n} \) (or \( \leq 2\sqrt{n} \))

last step uses \( \leq \sqrt{n} \) new colors

\( \Rightarrow \) total colors \( \leq 3\sqrt{n} \)
Max Cut (2 ways!)

Given: Graph \( G = (V, E) \)

Output: partition of \( V \) into \( V_1, V_2 \)
to maximize "outside" i.e. \((u, v) \in E\)
st. \( u \in V_1, v \in V_2 \)

decision problem: is there cut of size \( \geq k \)?
NP-hard

Approx. Algorithm 1

- place node 1 in \( V_1 \)
- Consider remaining nodes in arbitrary order: \( i_2, i_3, \ldots, i_n \)
  to place \( i_j \):
    - if node \( i_j \) has more edges to nodes in \( V_1 \) than to nodes in \( V_2 \)
      place \( i_j \) in \( V_2 \)
    - o.w. place \( i_j \) in \( V_1 \)

Runtime: \( O(n) \)

Performance: at every placement \( \# \text{edges crossing cut} \geq \# \text{edges not crossing} \)
\( \implies \geq \frac{n}{2} \text{edges cross cut} \leq \text{since max cut size is } n \)
this is 2-approximation!
Randomized Approx Alg 2

- for each node \( i \in V 
  - \text{flip coin}
  - \text{if } H_i, \text{ place } i \text{ in } V_1
    - \text{else (T) place } i \text{ in } V_2

\text{Runtime? } O(n)

\text{Performance?}

\[
E[\text{cut size}] = E \left[ \sum_{(u,v) \in E} 1_{(u,v)} \right]
\]

\[
= \sum_{(u,v) \in E} E[1_{(u,v)}] \quad \text{linearity of expectation}
\]

\[
= \sum_{(u,v) \in E} \Pr[(u,v) \text{ crosses cut}] \quad \text{expectation of indicator variable is prob it equals 1}
\]

\[
= \sum_{(u,v) \in E} \Pr[u \in V_1 \cup v \in V_2 \text{ or } u \in V_2 \cup v \in V_1]
\]

\[
= \sum_{(u,v) \in E} \Pr[u \in V_1 \cup v \in V_2] + \Pr[u \in V_2 \cup v \in V_1] \quad \text{union of disjoint events}
\]

\[
= \sum_{(u,v) \in E} \Pr[u \in V_1] \cdot \Pr[v \in V_2] + \Pr[u \in V_2] \cdot \Pr[v \in V_1] \quad \text{u+v's placement are independent events}
\]
\[
= \sum_{(u,v) \in E} \left( \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} \right)
\]
\[
= \sum_{(u,v) \in E} \frac{1}{2}
\]
\[
= \frac{|E|}{2}
\]

This analysis weaker than previous?

We expect a pretty good result, but what is probability that we get it?

(pretty good, but not shown here)
Metric Travelling Salesman Problem (Metric Tsp)

Given \( n \times n \) matrix \( w \) st. \( w(u,v) \geq 0 \)

\[ \Delta \Rightarrow w(u,v) \leq w(u,x) + w(x,v) \]

\( \forall u,v \in V \)

Problem find cycle \( V_1 \rightarrow V_2 \rightarrow \ldots \rightarrow V_n \rightarrow V_1 \)

visiting each node exactly once

+ minimizing \( \frac{1}{n} \sum_{i=1}^{n} w(v_i, v_{i+1}) \)

\( \sum_{i=1}^{n} w(v_i, v_{i+1}) \)

Approximation Algorithm:

1. Compute MST \( T \)
2. Pick arbitrary root node for \( T \)
3. Output preorder traversal of \( T \)

 equivalently:

- Walk "around" \( T \) (OS) visits each edge twice
- Modify walk to skip repeats of nodes by taking shortcut to next unseen node

\( \frac{1}{2} \) guarantee: not longer due to \( \Delta \)
example

\[
\begin{align*}
A & \quad 6 \\
B & \quad 1 \\
C & \quad 4 \\
D & \quad 5 \\
\end{align*}
\]

Preorder Traversal: 6

\[D \rightarrow C \rightarrow A \rightarrow \cancel{C} \rightarrow D \rightarrow B \rightarrow D\]

Skip repeats

length of preorder traversal = \(2 \cdot w(MST)\)

length of walk with shortcuts ≤ \(2 \cdot w(MST)\)

length of OPT TSP ≥ \(w(MST)\)

why? take OPT TSP cycle. remove any edge, gives spanning tree \(T\) with \(w(T) \leq w(\text{OPT TSP})\)

\(w(MST)\) is at most \(w(T)\)

\[w(\text{output}) \leq 2 \cdot w(MST) \leq 2 \cdot w(\text{OPT})\]

Christofides' algorithm gives 1.5 approximation.

For planar TSP, Arora/Mitchell give poly time approx to any \((1 + \epsilon)\) - factor.