lecture 10

Approximation Algorithms

\[ \text{REVIEW} \]

- Dynamic Programming
- Covering TSP
- Basic Graph Theory
  - Matching, TSP, MSTs, Hamiltonian/Euler circuits
  - Independent Sets, Cliques
- Applications of Max Flow
- Dynamic Method
- The probabilistic method

Reductions between [Vertex Cover, ]

**Dynamic Programming 3.0**

1. Find a recursive structure ([Find definition for problem])
   - Substructure: change problem a little.
   - \# subproblems
   - Time for subproblem

\[ \text{TSP} \]

- Consider Hamiltonian paths
- Add ending

2. Memoize:
   - Memory: \( O(\# \text{ of subproblems}) \)
   - Space: \( O(\text{memory} \times \text{amount of time used for subproblem}) \)
Let's say $n$ decisions to go down, $n$ decisions to go right.

for $n$ by $m$ grid $(n+m)$ ways to

Let's define $\text{Best}(i,j) = \max \text{ you can get starting at } (i,j) \text{ and ending at } (n,n)$

What is the base case?

Base $(n,n) = 3^n$

$$= \max \begin{cases} a_{i,j+1} + \text{Best}(i,j+1) \text{ (if } j+1 \text{ is in bound)} \medskip \\ a_{i+1,j} + \text{Best}(i+1,j) \text{ (if } i+1 \text{ is in bound)} \end{cases}$$

Final Answer: $a_{11} + \text{Best}(1,1)$
Clever way of doing it.

Fill in array yourself without recursion.
(Throw away parts of the memoization that aren't needed).

Covering Times

1. \( C(G) \leq 2m(n-1) \)
2. \( C(G) \leq 2mR_{uv} \ln(n) + n \)
   \[ = 0(mR\log(n)) \]

\( R(G) = \max_{u,v} R_{uv} \)

Ruv effective resistance.

Mathews Bound:

\( t \leq C(G) \leq H \log(n) \)

longest hitting time.

Hypercube Example:

- Each node is a bit string of length \( h \).
- There is an edge if and only if two bitstring differ in exactly one location.

Degree of every node is \( h \), \[ \deg(v) = h \]
\[ = 0(n^2\log(n)) \]
Even 1's

Odd's

A Independent Set of size $2^{n-1}$

Since bipartite only has a side bigger than $\frac{n}{2}$.

→ Prove $C(\mathbb{A}_n) \leq O(nh^3)$

What is the diameter of $\mathbb{A}_n = h$ (diameter is at most $h$).

$R(\mathbb{A}_n) \leq h$

→ $C(A) \leq O(h \cdot h \cdot \log(2^h) / h) = O(nh^3)$

Better $O(hn1/h) = O(nh^2)$

$log(2^h) = \text{order } h$

$log(2^h) = \text{order } h$ in Big $O$.

Show that max flow is at least $h$.

Show, there are

$000000\rightarrow 010000 \rightarrow 0000111$

We have parallel edges. Each path has resistance $\frac{1}{h}$.

$R(u) \leq \frac{1}{h} + \frac{1}{h} + \cdots + \frac{1}{h} = \frac{h}{h} = 1$

But bound in HW is more loose.

$O(n^2 \log(n)))$ not too tight.
2-SAT: Undirected chain \rightarrow \text{polynomial time algorithm}

3-SAT: directed chain \rightarrow \text{Exponential covering time}

Worth thinking about: Chain of Reductions?

\leq_p \text{SAT} \quad \text{(3-SAT is the hardest problem of all by Cook's Theorem)}

- If I could do Independent Set, I could also do 3-SAT

3-SAT \leq_p \text{Independent set}

\text{gadget construction}
\text{Relaxation for vertex cover}

\text{Vertex cover} \leq_p \text{Integer Programming}
\text{Relaxation for vertex cover}

\rightarrow \text{We also showed, if I could find, I could find vertex cover}

\text{Independent set} \leq_p \text{Vertex Cover}\text{ complement}

\text{SAT} \leq_p \text{3-SAT}
\text{by De Morgan's laws}