**Final Exam**

**Comprehensive Exam**
- 25% - 25% - 50%
- Part I

**Part I**
- Chapter 0, 1, 2
- Chap 0 \(\rightarrow\) big oh
- Chap 1 \(\rightarrow\) basic arithmetic operations (mul/div)
- exponentiation
- Euclid's GCD
- Primality testing
- Skip 1.4 \(\leftarrow\) crypto
- Hashing

**Chap 2 \(\rightarrow\) Divide & Conquer
- Selection \(\leftarrow\) randomized
- Sorting \(\leftarrow\) quicksort
- Master theorem
- Skip 2.6 \(\leftarrow\) FFT

**Part II**
- Chapters 3, 4, 5
- Chap 3 : DFS
- \(\rightarrow\) pre-order
- \(\rightarrow\) SCC/CC
- \(\rightarrow\) topological sort
- \(\rightarrow\) DAGs

**Chap 4 :**
- \(\begin{align*}
\text{BFS} & \quad \text{Dijkstra (PA)} \\
\text{Fleiman-Ford (negative)} & \quad \text{shortest path}
\end{align*}\)
Chaps: greedy algorithms

- Dijkstra (PQ)
- Bellman-Ford (negativ)

Chaps: shortest path

- MST
- Prim+PQ
- Kruskals + Union-find

Chaps: spanning trees

Part III

Chapters: 6, 7, 8, 9

Chaps: Dynamic Programming

- All pairs shortest path
- Shortest path in DAGs
- TSP
- Longest increasing seq
- Edit distance
- Knapsack

Chap 7

Skip 7.1

Sec 7.2

- Flows in Networks (max-flow/min-cut)
- Bipartite Matching

Skip rest: 7.4-7.7

Chap 8

- NP/PL-completeness
- Reduction
- Optimization

TSP | Hamilton | Path | SAT | Knapsack

Chap 9

Approximation Ratio

TSP (9.2.3)
**TSP**: find a tour that visits each city only once, start and end @ same city, given a complete graph G, with weight \( d_{ij} \) = distance between i and j.

NP-complete

Reduction from Rudrata cycle, using the approx ratio within the constant c.

**Hard to Approximate**

**Metric TSP**: TSP problem + one additional property

\[ G, d_{ij} \quad \forall i, j \in V \]

\[ d_{ab}, d_{ac}, d_{bc} \]

\[ d_{ik} \leq d_{ij} + d_{jk} \]

\[ d_{ac} \leq d_{ab} + d_{bc} \]

Approximate with ratio 2.
Metric TSP: Approximation Algorithm

Given G, \( d_{ij} \) that satisfy triangle inequality.

\[ A(G) : \]
1. Compute the MST for G

\[ \alpha = \max G \quad A(G) \leq 2 \leq \frac{1}{\text{opt-TSP}} \]

\[ \alpha = \frac{A(I)}{\text{opt-TSP}} \leq \frac{A(I)}{\text{MST}} \]

\[ \text{MST} \leq \text{opt-TSP} \cdot \sum_{i=0}^{n-2} d_{i,i+1} \]

2) Create a double walk:
\[ a \rightarrow b \rightarrow d \rightarrow [e \rightarrow d \rightarrow c] \rightarrow d \rightarrow [b \rightarrow c] \rightarrow a \]

Cost: \( 2 \cdot \text{MST} \)

3) Transform into a tour:
\[ a \rightarrow b \rightarrow d \rightarrow [e \rightarrow c] \rightarrow a \]

\[ d_{ec} \leq d_{ed} + d_{ec} \leq \text{triangle inequality} \]

\[ A(I) \leq 2 \cdot \text{MST} \]
\[ A(I) \leq 2 \text{MST} \]

\[ \alpha \leq \frac{A(I)}{\text{MST}} \leq \frac{2\text{MST}}{\text{MST}} = 2 \]

In polynomial time, we can approximate metric TSP.

Q9.7: Given a graph \( G \), given distinct vertices \( s_1, s_2, s_3, \ldots, s_k \), partition \( G \) into \( k \) parts, such that each \( s_i \) belong to a different part, minimize the cut edges between the different parts.

Multiway minimum cut

\[ u \leq \gamma \]

\[ \text{Cut} : \text{the edges that cross from one part to another} \]

Find min \( k \)-way cut

1) When \( k = 2 \), give a polynomial time solution

\[ s_1 = s, \quad s_2 = t \]

same

sink

Find a max-flow in \( G \) from \( s \) to \( t \)

Extract the minimum
b) give an approximation for \( k = 3 \), approx ratio \( \leq 2 \)

Optimal 3-way cut, \( C^* \), let the cut be \( E^* \)

Compute the optimal

1) \((s_1, s_2)\) cur (minor) \( E_{12} \) # g cross cuts

2) \((s_2, s_3)\) minor \( E_{23} \) # cross cuts

3) return union of edge in \( E_{12} \) and \( E_{23} \)

\[ |E_{12}| \leq |E^*| \]
\[ |E_{23}| \leq |E^*| \]
\[ |E_{12} \cup E_{23}| \leq |E_{12}| + |E_{23}| \leq 2|E^*| \]

\[ \alpha = \frac{A(I)}{E^*} \leq \frac{2|E^*|}{|E^*|} = 2 \]

---

Q 5.13: Give a polynomial solution or show that it is \( \text{NP} \)-complete

given undirected graph \( G \)

a) given a set \( L \subseteq V \), find a spanning tree such that its leaves include the node in \( L \)

\[ L = \{a, c\} \]

Find any spanning tree that
1) delete an arc from $G$ to obtain $G' = V = L$
2) find a spanning tree in $G'$, say $T'$
3) add back any edge that connects vertices in $L$ to $T'$

Polynomial time $O(|V| + |E|)$

b) find a spanning tree, but the leaves must be exactly $L$

\underline{Yudrata problem (s-t)}

$G$, let $L = \{s, t\}$

leaves must be exactly $s$ and $t$

(only 2 leaves are allowed in the spanning tree)

Any tree that has exactly 2 leaves, must be a path

If $G$ has a spanning tree with $L = \{s, t\}$ as the exact 2 leaves
then that tree is a path that
visits all the vertices exactly once

$\Rightarrow$ Yudrata s-t path
c) Similar to b.

d) Find a spanning tree with $k$ or fewer edges.

$\text{NP-complete}$

Since when $k=2$, this is the same as $\text{Rudraka set path}$.