CMPUT 498/501: Advanced Algorithms **Fall 2025**

Lecture 1 (Sep 4, 2025): Template file

Lecturer: Mohammad R. Salavatipour *Scribe: Mohammad R. Salavatipour*

1.1 Introduction

You can write stuff here. This document does not have a coherent content and is intended to be used to see some examples of using Latex for typesetting mathematical forumals and content. So things will look out of place as they are borrowed from various (irrelevant) places.

Recall that P is the class of problems solvable in polynomial time and NP (informally) are those (decision) problems whose solutions can be verified by a polynomial time algorithm.

Here is how to create numeric itemized list:

1. first item,
2. second item,
3. and third item

We can talk about $\text{P} \neq \text{NP}$ or polynmial time algorithms. Here is a different itemized list with reference to the previous one:

- relax (3), then we are into study of special cases of the problem.
- relax (2), we will be in the field of integer programming and the techniques there such as branch-and-bound, etc.
- relax (1), we are into study of heuristics and approximation algorithms.

1.1.1 A Linear Program

Consider linear program (**TSP-LP**) below.

$$\begin{aligned}
\text{minimize:} \quad & \sum_e c(e) \cdot x_e & & & \textbf{(TSP-LP)}\\
\text{subject to:} \quad & x(\delta(S)) \geq & 2 & \quad \text{for each cut } \emptyset \subsetneq S \subsetneq V & (1.1)\\
& x(\delta(v)) = & 2 & \quad \text{for each vertex } v \in V & (1.2)\\
& x \geq & 0 & &
\end{aligned}$$

Constraints (1.1) are the *cut constraints* and Constraints (1.2) are the *degree constraints*.

1.1.2 Tips

Use $\log n$, not $log\ n$.

$V = \{v_1, v_2, \ldots, v_n\}$.

Check out $\sum_{i=1}^n i$ vs. $\displaystyle\sum_{i=1}^n i$.

A displayed equation:

$$H_n = \sum_{k=1}^n \frac{1}{k} = \int_1^n \frac{dx}{x} + O(1) = \ln n + O(1)$$

A matrix:

$$\begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ 0 & 0 & 1 \end{pmatrix}$$

Problem names should look like this: SET COVER.

Multiple lines under a sum:

$$\sum_{\substack{k=1 \\ k \text{ odd}}}^n k = \left\lceil \frac{n}{2} \right\rceil^2$$

1.1.3 An Algorithm

Algorithm 1 describes Kruskal's MINIMUM SPANNING TREE procedure. Notice how the algorithm is not appearing right after this paragraph even though it does in the `.tex`. This is ok for scribe notes, the LaTeXcompiler will still put it close.

Algorithm 1 Kruskal's MINIMUM SPANNING TREE Algorithm

Input: Undirected graph $G = (V, E)$ with edge costs $c(e) \geq 0, e \in E$.
Output: A minimum spanning tree of G.

```
  $T \leftarrow \emptyset$
  for each edge $e \in E$ in increasing order of cost $c(e)$ do
    if $T \cup \{e\}$ does not contain a cycle then
      $T \leftarrow T \cup \{e\}$
    end if
  end for
  return $T$
```

1.2 NP Optimization problems

An NP optimization problem, Π, is a minimization (maximization) problem which consists of the following items:

Valid instances: each valid instance I is recognizable in polynomial time. (note: 'Polynomial time' means polynomial time in terms of the size of the input.) The set of all valid instances is denoted by D_Π. The size of an instance $I \in D_\Pi$, denoted by $|I|$, is the number of bits required to represent I in binary.

Feasible solutions: Each $I \in D_\Pi$ has a set $S_\Pi(I)$ of feasible solutions and for each solution $s \in S_\Pi(I)$, $|s|$ is polynomial (in $|I|$).

Objective function: A polynomial time computable function $f(s, I)$ that assigns a non-negative rational value to each feasible solution s for I.

We often have to find a solution s such that this objective value is minimized (maximized). This solution is called optimal solution for I, denoted by $OPT(I)$.

Examples:

- Vertex Cover: In this case we have:

  valid instances : set of graphs with weighted vertices.

  feasible solutions : all the vertex covers of the given graph.

  objective functions : minimizing the total weight of a vertex cover.

- Minimum Spanning Tree (MST) problem: Given a connected graph $G(V, E)$, with each edge $(u, v) \in E$ assigned a weight $w(u, v)$, find an acyclic subset $T \subseteq E$ that connects all the vertices and its total weight is minimized. Since T is acyclic and connects all of the vertices, it is a tree.

  valid instances : a graph with weighted edges.

  feasible solutions : all the spanning trees of the given weighted graph.

  objective functions : minimizing the total weight of a spanning tree.

1.3 Approximation algorithms

An *α-factor approximation algorithm* (or simply an α-approximation) is a polynomial time algorithm whose solution is always within α factor of optimal solution.

Definition 1 *For a minimization problem Π, algorithm A has approximation factor α if it runs in polynomial time and for any instance $I \in D_\Pi$ it produces a solution $s \in S_\Pi(I)$ such that $f(s, I) \leq \alpha(|I|) \cdot OPT(I)$. α can be a constant or a function of the size of the instance.*

We use $A(I)$ to denote the value of the solution returned by algorithm A for instance I; therefore, from Definition 1: $A(I) \leq \alpha(|I|) \cdot OPT(I)$. Below are some theorems and lemmas with complex formulas.

Theorem 1 *Assume that $\mathcal{F}$ and $\mathcal{E}$ are defined as above. There exists a constant δ such that for any $0 < \epsilon \leq 1$ the following holds:*
Suppose that every trial $f_i \in \mathcal{F}$ has a constant number of outcomes and we can carry out the random trial in time t_1. Let $p_i = e^{-\epsilon|F_i|}$ and suppose that for all $S \subseteq F_i$ it holds that:

- *if $|S| > \epsilon|F_i|$ then $\Pr[E_i|_S] \leq p_i$,*
- *if $|S| \leq \epsilon|F_i|$ then $\Pr[E_i|_S] = 0$, and*

- *knowing the outcomes of the trials in S, we can evaluate whether $E_i|_S$ holds or not in time t_2.*

Furthermore assume that for $x_i = e^{-\delta\epsilon^2|F_i|}$ it holds that $x_i \leq \frac{1}{e}$ and:

$$p_i^\epsilon \leq x_i \prod_{E_j \in N(E_i)} (1 - x_j), \qquad 1 \leq i \leq m. \tag{1.3}$$

Then there is a randomized algorithm that finds the outcomes of the random trials in time $O((t_1+t_2)\times \mathrm{Poly}(n+m))$ with high probability, where $\mathrm{Poly}(n+m)$ is a polynomial in $(n+m)$, such that for every event E_i, the set F_i is partitioned into at most 3 subsets $S_{i,1}, S_{i,2}, S_{i,3}$, so that $E_i|_{S_{i,1}}$, $E_i|_{S_{i,2}}$, and $E_i|_{S_{i,3}}$ are all false.

Now our goal is to prove the following lemma, by which the main lemma can be proved easily.

Lemma 1 *The expected number of (1,2)-trees T with order at least Ψ is at most $21me^{-\Psi/20}$.*

For a possible (1,2)-tree T, we say T starts from E_0, if E_0 is the initial event of the first 1-component of T. Define

$$\mathcal{T} = \{\text{possible (1,2)-trees } T \text{ with order } O_T = \Psi \text{ that start at } E_0\}.$$

Now let $\mathcal{T}' \subseteq \mathcal{T}$ be the set of (1,2)-trees obtained after Step 1 of the algorithm that are also in $\mathcal{T}$. By this definition and (**??**):

$$E[|\mathcal{T}'|] = \sum_{T\in\mathcal{T}} \Pr[Z_T] \leq \sum_{T\in\mathcal{T}} \prod_{E_j : v_j \in V_C(T)} p_j. \tag{1.4}$$

$$\begin{aligned} E[|\mathcal{T}''_{S_1,\ldots,S_k}|] &= e^{-S_0} \sum_{\substack{\text{all } l_1\text{'s} \\ O_{l_1}=S_1}} \sum_{\substack{\text{all } l_2\text{'s} \\ O_{l_2}=S_2}} \cdots \sum_{\substack{\text{all } l_k\text{'s} \\ O_{l_k}=S_k}} \prod_{t=1}^{k} \prod_{E_j \in l_t} p_j \\ &= e^{-S_0} \sum_{\substack{\text{all } l_1\text{'s} \\ O_{l_1}=S_1}} \Big(\prod_{E_{j_1}\in l_1} p_{j_1} \sum_{\substack{\text{all } l_2\text{'s} \\ O_{l_2}=S_2}} \Big(\prod_{E_{j_2}\in l_2} p_{j_2} \cdots \sum_{\substack{\text{all } l_k\text{'s} \\ O_{l_k}=S_k}} \prod_{E_{j_k}\in l_k} p_{j_k} \Big)\Big) \ldots \Big). \end{aligned} \tag{1.5}$$

Denote the set of all extensions R with $O_R = r$ of a set Q by $\mathrm{EXT}(Q,r)$. For a set Q of events let $X_{Q,r}$ be the number of extensions R such that $R \in \mathrm{EXT}(Q,r)$ and all events in R have heads tags. Therefore:

$$E[X_{Q,r}] = \sum_{R:R\in\mathrm{EXT}(Q,r)} \Pr[Z'_R] = \sum_{R:R\in\mathrm{EXT}(Q,r)} \prod_{E_j\in R} p_j. \tag{1.6}$$

By this equation, the most internal summation in (1.5) is in fact $E[X_{l_{k-1},S_k}]$. In Section **??**, Lemma **??**, we show that $E[X_{Q,r}] \leq e^{-r/8} e^{O_Q/16}$. Therefore, $E[X_{l_{k-1},S_k}] \leq e^{-S_k/8} e^{S_{k-1}/16}$. Using this fact, Inequality (1.5) can be written as:

$$E[|\mathcal{T}''_{S_1,\ldots,S_k}|] \leq e^{-S_0} \prod_{i=0}^{k-1} e^{-S_{i+1}/8} e^{S_i/16}. \tag{1.7}$$

We need the following combinatorial lemma to prove Lemma 1.

Lemma 2 *Let N_x^+ denote the set of integers greater than or equal to x. For $1 \leq k \leq \delta\psi$, define EQ_k to be the equation $S_1 + S_2 + \ldots + S_k = \psi$, where the domain of each variable S_i $(1 \leq i \leq k)$ is $N^+_{\frac{1}{\delta}}$. Then for sufficiently small $\delta > 0$, the sum of the number of the solutions of all equations EQ_k $(1 \leq k \leq \delta\psi)$ is at most $e^{\psi/80}$.*

Proof. Let's call the equation $r_1 + r_2 + \ldots + r_{\delta\psi} = \psi$, in which the r_i's are the variables whose domain is N_0^+, the *reference* equation. To each solution of equation EQ_k, for $1 \leq k \leq \delta\psi$, we associate a unique solution of the reference equation: set $r_i = S_i$, for $1 \leq i \leq k$, and $r_j = 0$, for $k < j \leq \delta\psi$. Therefore, the sum of the number of solutions of all EQ_k equations (for $1 \leq k \leq \delta\psi$) with domain N_k^+, is not more than the number of solutions of the reference equation with domain N_0^+. From elementary combinatorics we know that the number of non-negative integer solutions of the reference equation is $\binom{\psi+\delta\psi-1}{\delta\psi-1}$, which is less than $\binom{\psi+\delta\psi}{\delta\psi}$. Using Stirling's approximation for $n!$:

$$\begin{aligned}
\binom{\psi+\delta\psi}{\delta\psi} &\leq \frac{[\psi(1+\delta)]^{\psi+\delta\psi}}{(\delta\psi)^{\delta\psi}\psi^{\psi}} \\
&\leq \frac{e^{\delta\psi}(1+\delta)^{\delta\psi}}{\delta^{\delta\psi}} \\
&\leq e^{\delta\psi(1+\ln(1+\frac{1}{\delta}))} \\
&\leq e^{\psi/80}
\end{aligned}$$

if δ is sufficiently small. ■

Proof of Lemma 1: Using (1.7), definition of $\mathcal{T}''_{S_1,\ldots,S_k}$, and Lemma 2:

$$\begin{aligned}
E[|\mathcal{T}''|] &\leq \sum_{k=1}^{\delta\psi} \sum_{\substack{\frac{1}{\delta} \leq S_1,\ldots,S_k \leq \psi \\ \sum_{1\leq i\leq k} S_i = \psi}} E[|\mathcal{T}''_{S_1,\ldots,S_k}|] \\
&\leq e^{-S_0} \sum_{k=1}^{\delta\psi} \sum_{\substack{\frac{1}{\delta} \leq S_1,\ldots,S_k \leq \psi \\ \sum_{1\leq i\leq k} S_i = \psi}} \prod_{j=0}^{k-1} e^{-S_{j+1}/8} e^{S_j/16} \\
&\leq e^{-S_0} e^{-\psi/8} e^{\Psi/16} \sum_{k=1}^{\delta\psi} \sum_{\substack{\frac{1}{\delta} \leq S_1,\ldots,S_k \leq \psi \\ \sum_{1\leq i\leq k} S_i = \psi}} 1 \\
&\leq e^{-\Psi/8} e^{\Psi/16} e^{\Psi/80} \\
&= e^{-\Psi/20} \qquad (1.8)
\end{aligned}$$

for sufficiently small δ. Therefore, using (1.8):

$$E[|\mathcal{T}'|] \leq e^{-\Psi/20}. \qquad (1.9)$$

Since we have at most m events that can be the initial event of a (1,2)-tree T with total order at least Ψ, using the bound in (1.9), after Step 1 of the algorithm:

$$\begin{aligned}
E[|\{\text{(1,2)-trees } T \text{ of order at least } \Psi\}|] &\leq m \sum_{k \geq \Psi} e^{-k/20} \\
&\leq 21 m e^{-\Psi/20}.
\end{aligned}$$

■

Here is another big formula:

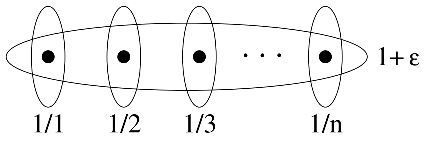

Figure 1.1: A tight example for $SC1$.

$$\begin{aligned}
x_{c,j} \prod_{F_j \cap F_t \neq \emptyset} (1 - x_{s,t}) &\geq e^{-\delta\epsilon^3|e_j|} \prod_{k \geq 1/\lambda} \left(1 - e^{-\delta\epsilon^3 k}\right)^{C(\beta|e_j|e^{\gamma k}+1)} \\
&\geq e^{-\delta\epsilon^3|e_j|} \exp\left\{-2\beta C'|e_j| \sum_{k \geq 1/\lambda} e^{-\delta\epsilon^3 k} e^{\gamma k}\right\} \\
&\qquad \text{(For } C' = C+1 \text{ and sufficiently small } \lambda) \\
&\geq \exp\left\{\left[-\delta\epsilon^3 - \left(\frac{2\beta C' a^{1/\lambda}}{1-a}\right)\right]|e_j|\right\} \qquad \text{(where } a = e^{\gamma - \delta\epsilon^3}) \\
&\geq e^{-\epsilon^3|e_j|} \qquad \text{(if } \beta \text{ and } \gamma \text{ are sufficiently small)} \\
&= p_{c,j}^{\epsilon^2}.
\end{aligned}$$

Here is how to include a figure at the top of the page (see Figure 1.1.)

Or you can have your figure go right here (see Figure 1.2)

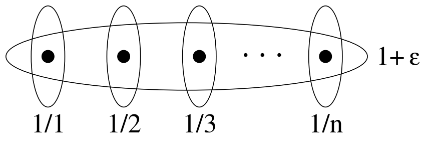

Figure 1.2: A tight example for $SC1$.