Lecture 1

Sep. 5 2024

1. Asymptotic Notations.

Θ: $f(n) = \Theta(g(n))$.

$\exists\, c_1, c_2, n_0$, s.t. $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$, $\forall n \ge n_0$.

O: $f(n) = O(g(n))$.

$\exists\, c, n_0$, s.t. $0 \le f(n) \le c \cdot g(n)$, $\forall\, n \ge n_0$.

o: $f(n) = o(g(n))$.

$\forall\, c > 0$, $\exists\, n_0$, s.t. $0 \le f(n) < c \cdot g(n)$, $\forall n > n_0$. (严格更小).

e.g. $n \log\log n = o(n \log n)$

Ω: $f(n) = \Omega(g(n))$.

$\exists\, c, n_0$, s.t. $0 \le c \cdot g(n) \le f(n)$, $\forall\, n > n_0$.

ω: $f(n) = \omega(g(n))$.

$\forall\, c > 0$, $\exists\, n_0$, s.t. $0 \le c \cdot g(n) < f(n)$, $\forall n > n_0$.

$f(n) = o(g(n))$ iff. $g(n) = \omega(f(n))$.

2. Solving Recurrences.

(1) Guess and Prove by Induction.

$T(n) = 2T(n-1) + 1$.

$T(0) = 0$, $T(1) = 1$, $T(2) = 3$, $T(3) = 7$, ...

Guess: $T(n) = 2^n - 1$.

Proof: $T(0) = 2^0 - 1 = 0$ Base case.

$T(n) = 2T(n-1)+1$
$= 2(2^{n-1}-1)+1$ ↙ by IH.
$= 2^n - 1$.

$F_n = F_{n-1} + F_{n-2}$, $F_0 = 0$, $F_1 = 1$.

Guess: $F_n \le \alpha \cdot c^n$, $\alpha > 0$, $c > 1$ (Upper bound)

Proof (and solve α, c):

$F_n = F_{n-1} + F_{n-2}$
$\le \alpha c^{n-1} + \alpha c^{n-2}$
$\overset{?}{\le} \alpha c^n$
$\Leftrightarrow$
$c^{n-1} + c^{n-2} \le c^n$
$\Leftrightarrow$
$1 + c \le c^2$ ..... φ_+ satisfies this ineq

this ineq stands for $c = \varphi_+$.
$\Rightarrow$ inductive step valid.

$c^2 - c - 1 = 0 \Rightarrow \varphi_+ = \frac{1+\sqrt{5}}{2}$, $\varphi_- = \frac{1-\sqrt{5}}{2}$.

$F_n \le \alpha \cdot \varphi_+^n$.

According to base case, $F_0 = 0 \le \alpha \cdot \varphi_+^0 \Rightarrow \alpha \ge 0$
$F_1 = 1 \le \alpha \cdot \varphi_+^1 \Rightarrow \alpha \ge \frac{\sqrt{5}-1}{2} \approx 0.618$.

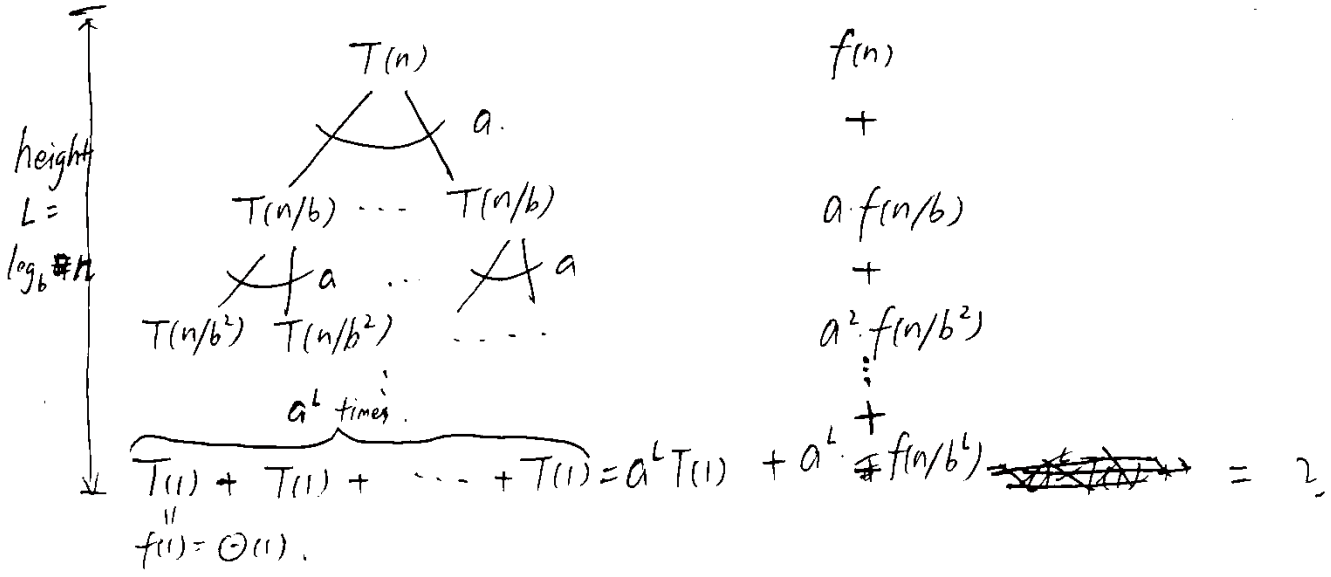
$F_n \le \alpha \cdot c^n \le \frac{1}{\varphi_+} \cdot \varphi_+^n = \varphi_+^{n-1}$.

Guess: $F_n \ge \beta \cdot \varphi^n$ (Lower bound)
用和上界一模一样的方法, $F_n \ge \varphi_+^{n-2}$.

Conclude: $\varphi_+^{n-2} \le F_n \le \varphi_+^{n-1}$.

(2) Devide-and-Conquer (Recursion Trees).

$T(n) = a \cdot T(n/b) + f(n)$.



Example.

$T(n) = T(\frac{3}{4}n) + n$. $a=1$, $b=\frac{4}{3}$, $f(n)=n$.

level i: $a^i \cdot f(n/b^i) = 1 \cdot \frac{n}{(4/3)^i} = (\frac{3}{4})^i n$.

$L = \log_{4/3} n$.

$T(n) = \sum_{i=0}^{L} (\frac{3}{4})^i n = \Theta(n)$.

Example.

$T(n) = 3T(n/2) + n$. $a=3$, $b=2$, $f(n)=n$.

level i: $a^i \cdot f(n/b^i) = 3^i \cdot n/2^i = (\frac{3}{2})^i \cdot n$, $L = \log_2 n$.

$T(n) = \sum_{i=0}^{L} (\frac{3}{2})^i n = \Theta(n^{\log_2 n})$.

Example.

$T(n) = 2T(n/2) + \frac{n}{\log_2 n}$. $a=2$, $b=2$, $f(n) = n/\log_2 n$.

cost incurred at level i: $a^i f(n/b^i) = 2^i \cdot \frac{n/2^i}{\log_2(n/2^i)} = \frac{n}{\log_2 n - \log_2 2^i} = \frac{n}{\log_2 n - i}$.

$L = \log_2 n$.

$T(n) = \sum_{i=1}^{L} \frac{n}{\log_2 n - i}$, 令 $x = \log_2 n$. $T(n) = n \cdot (\frac{1}{x} + \frac{1}{x-1} + \frac{1}{x-2} + \cdots) \sim O(n \log \log n)$

Harmonic sequence.

(3). Akra - Bazzi Theorem.

$T(n) = \sum_{i=1}^{k} a_i \cdot T(n/b_i) + f(n)$, k, a_i, b_i are constants.

$\forall i$, $f(n) = \Omega(n^c)$ and $f(n) = O(n^d)$, i.e., $c_2 n^c \le f(n) \le c_1 n^d$.

then, $T(n) = \Theta\left(n^\rho \left(1 + \int_1^n \frac{f(u)}{u^{\rho+1}}\, du\right)\right)$, ρ: real solution to $\sum_{i=1}^{k} \frac{a_i}{b_i^\rho} = 1$.

Example. $T(n) = T(\frac{3}{4}n) + T(\frac{1}{4}n) + n$.

$(\frac{3}{4})^\rho + (\frac{1}{4})^\rho = 1 \Rightarrow \rho = 1$.

$T(n) = \Theta\left(n^1 \cdot \left(1 + \int_1^n \frac{u}{u^{1+1}}\, du\right)\right) = \Theta\left(n^1 \cdot (1 + \log n)\right) = \Theta(n \log n)$.

Example. $T(n) = T(\frac{1}{5}n) + T(\frac{7}{10}n) + n$.

$(\frac{1}{5})^\rho + (\frac{7}{10})^\rho = 1 \Rightarrow 0 < \rho < 1$.

$\int_1^n \frac{f(u)}{u^{1+\rho}}\, du = \int_1^n \frac{u}{u^{1+\rho}}\, du = \int_1^n u^{-\rho}\, du = \frac{u^{1-\rho}}{1-\rho}\Big|_1^n = \frac{n^{1-\rho} - 1}{1-\rho} = \Theta(n^{1-\rho})$

$T(n) = \Theta(n^\rho (1 + n^{1-\rho})) = \Theta(n^\rho + n) = \Theta(n)$.

Example. $T(n) = T(n/2) + T(n/4) + 1$.

$(\frac{1}{2})^\rho + (\frac{1}{4})^\rho = 1 \Rightarrow \rho = \log_2 \frac{1+\sqrt{5}}{2}$.

$\int_1^n \frac{1}{u^{1+\rho}}\, du = \frac{u^{-\rho}}{-\rho}\Big|_1^n = \frac{1 - n^{-\rho}}{\rho} = \Theta(1)$.

$T(n) = \Theta(n^\rho (1 + \Theta(1))) = \Theta\left(n^{\log_2 \frac{1+\sqrt{5}}{2}}\right)$.

(4) Generating Functions. (Linear Recurrence).

Definition. $\{a_n\}_{n \ge 0}$ a sequence. $A(x) = \sum_{n \ge 0} a_n x^n$

Toss coin. head: p. tail $q = 1-p$. toss k times.

$a_n = \Pr[\#\text{head up} = n] = \binom{k}{n} p^n q^{k-n}$.

$$A(x) = \sum_{n=0} \binom{k}{n} p^n q^{k-n} x^n = (q+px)^k = \underbrace{(q+px)\cdot \dots (q+px)}_{k \text{ times.}}$$

一个复杂的 GF 可由多个简单的 GF 构成.

Example. diff. 6 × 1 cent, 1 × 5 cents, 2 × 1 dime. coins.
How many ~~values~~ money sums can be formed by combination?
1-cent coins: $A(x) = (1 + x + x^2 + x^3 + x^4 + x^5 + x^6)$. x^i: take i cents.
5-cents coins: $B(x) = (1 + x^5)$
1-dime coins: $C(x) = (1 + x^{10} + x^{20})$

$G(x) = A(x)B(x)C(x)$.
Expand. power of term: value (how many cents).
coefficient of term: how many ways of combinations.

Example. $f_n = f_{n-1} + f_{n-2}$, $f_0 = 1$.

$$F(x) = \sum_{i=0}^{\infty} f_i x^i.$$

$$F(x) = 1 + x + 2x^2 + 3x^3 + 5x^4 + 8x^5 + \dots$$
$$xF(x) = \quad x + x^2 + 2x^3 + 3x^4 + 5x^5 + \dots$$
$$x^2F(x) = \quad x^2 + x^3 + 2x^4 + 3x^5 + \dots$$

$\Rightarrow F(x) = xF(x) + x^2F(x) + 1$.
$\Rightarrow F(x) = \frac{1}{1-x-x^2} = \frac{a}{1-\varphi_+ x} + \frac{b}{1-\varphi_- x}$. φ_+, φ_-: roots of $1-x-x^2=0$

$$F(x) = a(1 + \varphi_+ x + \varphi_+^2 x^2 + \dots) + b(1 + \varphi_- x + \varphi_-^2 x^2 + \dots)$$
$$= (a+b) + (a\varphi_+ + b\varphi_-)x + (a\varphi_+^2 + b\varphi_-^2)x^2 + \dots.$$

$$\begin{cases} a+b=1 \\ a\varphi_+ + b\varphi_- = 1 \end{cases} \Rightarrow \begin{aligned} a &= \frac{1}{\sqrt{5}}\varphi_+ \\ b &= -\frac{1}{\sqrt{5}}\varphi_- \end{aligned}$$

Sep. 12 2024

Lecture 2

1. Minimum Spanning Tree (MST)

1.1 Problem Formulation.

Given an undirected connected graph $G=(V,E)$ w/ n nodes, m edges w/ weight $w(e)\in\mathbb{R}$.

Goal: Find a spanning tree / forest (if G disconnected). w/ minimum weight

Definition (spanning tree). A spanning tree of G is an acyclic subgraph T of G that is inclusion-wise maximal, that is, adding any edge in $G-T$ to T would create a cycle.

History

1.	1926.	Boruvka, parallel	$O(m\log n)$
2.	1930.	Jarnik, 1957, Prim	
3.	1956.	Kruskal	
4.	1975	Andrew C.-C. Yao.	$O(m\log\log n)$
5.	1984	Fredman & Tarjan	$O(m\log^* n)$
6.	1995	K K I	$O(m+n)$, randomized algorithm.
7.	1997.	Chazelle	$O(m\alpha(n))$, deterministic.

Definition (Operator $*$). Given operator f and constant c (usually take 3), $f^*(n)$ is the number of times that f have to be applied on n to make the result $<c$. That is, $\underbrace{f(f(\cdots f}_{k\text{ times}}(f(n))\cdots))<c$ (but $k-1$ f's on $n>c$), then

$f^*(n)=k$.

$\alpha(n)$: $\log^{**\cdots*}_{k\text{ times}} n < c$, then $\alpha(n)=k$.

Assumptions:

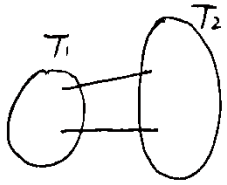
~~1)~~ (HW).

1) All edge weights are unique $\Rightarrow$ MST unique

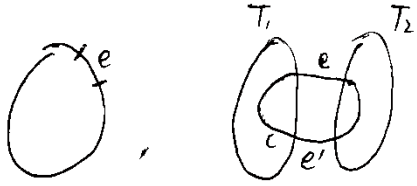
2) Graph is simple, i.e., $m\le\binom{n}{2}$.

1.2 Cut Rule & Cycle Rule.

Cut Rule. For any cut of the graph, the minimum-weight edge that crosses the cut must be in the MST.

Proof. 

Cycle Rule. For any cycle in G, the heaviest edge on the cyde cannot be in MST.

Proof. 

假定 e 在 T 中，则若删去 e，T就被分为 T_1、T_2 两部分。而 e 在 cycle C 中，这样一定有 $e' \in C$ 连接了 T_1、T_2. $T_1 + T_2 - e + e'$ 就构成了一棵新的生成树。但由e在C中 weight 最大，$w(e') < w(e)$，与T最小矛盾。 □

1.3 Kruskal Algorithm, 及 Jarnik / Prim Algorithm. & Bovuvka. Algorithm.

Kruskal.

① sort all edges by weights in ascending order s.t. $w(e_1) < w(e_2) < \cdots < w(e_m)$.

② iterate all edges in sorted order.
color an edge blue iff. it connects two vertices which are NOT currently in the same blue component. (不在同一个蓝色连通分支)
端点

Membership test uses: disjoint set with union-find, w/ operations:
1). makeset (elem) 2) find (elem) (返回 elem所在树根节点走, $\alpha(n)$ 时间).
3) union (elem1, elem2). $\alpha(n)$ time.

Analysis of kruskal:

① $O(m \log m)$. $n = 2m \Rightarrow O(m \log n)$

② i) 2 'find's for the two vertices of each edge. $m\alpha(n)$.
ii) $(n-1)$ 'union's. $(n-1)\alpha(n)$

$\Rightarrow T(n) = O(m \log n) + O(m\alpha(n)) + O((n-1)\alpha(n)) = O(m \log n)$.

Jarnik / Prim.

① take an arbitrary vertex x to start a T(blue)

② at each iteration, take the cheapest edge connecting T to some vertex NOT in T and color it blue

Finding cheapest edge: priority queue.

$m \times$ decrease-key, $O(1)$

$n \times$ extract-min.: $O(\log n)$

$\Rightarrow O(m \log n)$.

Boruvka

- Add many edges in parallel
- No non-trivial data structures.

① in each round

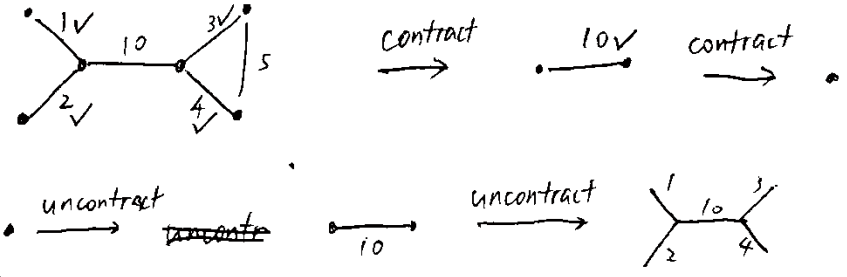
i) take cheapest edge out of each vertex and color it in blue.

ii) Contract all blue edges $a \overset{e}{\underset{\text{(blue)}}{\bullet\!-\!\bullet}} b \rightarrow \overset{\bullet}{ab}$ (e contracted, eliminated) form a smaller graph

iii) recurse (on the contracted graph)

② at the end, uncontract all blue edges to get the MST.

Example. tick ✓: colored blue.



Notice. 第一次 contract 时，3、4蓝了但5没有. contract 后·5就消失 (simple graph, 无自环). 反回来 uncontract 时 5也不会恢复，因为未被 标蓝.

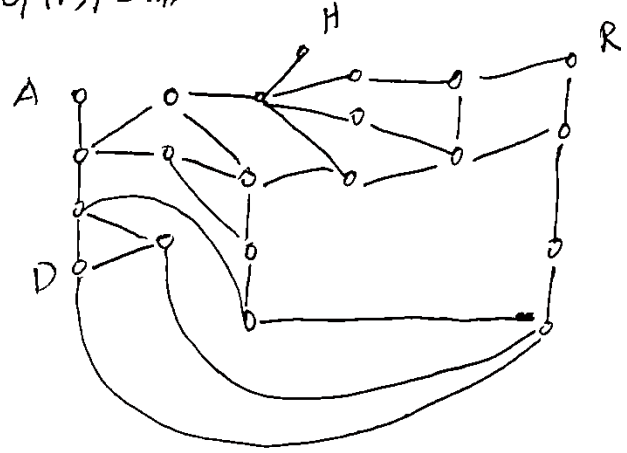
$O(m) \cdot \lceil \log_2 n \rceil$ $O(m \log n)$.

1.4 Fredman & Tarjan Algorithm

~~对第一次~~

For each iteration ~~for~~ i, maintain a threshold value k_i and run Prim algorithm. Once $|\text{NeighboursOf}(T)| \geq k_i$, choose another vertex and restart a Prim thereon.

Example.



Fredman & Tarjan.

① take an arbitrary vertex r to start a T (blue).

② ~~/~~ at each ~~iteration~~ recursion, take the cheapest edge connecting T to some vertex NOT in T and color it in blue

③ if $|N(T)| \geq k$ ($N(T)$: all neighbors of T) or if T has added an edge to some vertex that was marked, stop and mark all vertices in the current T. goto ①

④ terminate when all vertices are marked.

⑤ contract & recurse.

Each recursion: $n \times$ insert, $O(1)$

$(n-1) \times$ extract-min, ~~O(k)~~ $O(\log k)$

$n \times$ decrease-key, $O(1)$.

$\Rightarrow O(m + n \log k_2)$.

Claim. Every marked vertex u belongs to a connected component C s.t. $\sum_{v \in C} d_v \geq k$, d_v: degree of v in current graph. (graphs can be contracted by step ⑤).

Proof. Case. $|N(T)| \geq k$.

case. T has added an edge to some marked vertex.

□

$2m = \sum_{v \in V} d_v = \sum_{i=1}^{l} \sum_{v \in C_i} d_v \geq \sum_{i=1}^{l} k = kl$, m: # of edges in the current graph

$\Rightarrow l \leq \sqrt{\frac{2m}{k}}$, let $k_i = 2^{\lceil 2m/n_i \rceil}$

$\Rightarrow$ each recursion: $O(m_i + n_i \log k_i) = O(m_i \cdot n_i \cdot \frac{2m_i}{n_i}) = O(m_i)$

n_{i+1} = # of trees (connected components) in round i.

$n_{i+1} = f \le \frac{2m}{k_i}$, ~~$k_{i+1} = 2^{m/n_i} \ge 2^{m/2m/k_i}$~~ $k_i \le 2m/n_{i+1} = \log k_{i+1}$.

$k_{i+1} \ge 2^{k_i}$. $k_0 = 2^{2^{\cdots^{2}}}$ (0 times) $= n$, i.e., $\underbrace{\log \log \cdots \log n}_{0 \text{ times}} = c$

$\Rightarrow$ $\log^* n$ recursions. ~~$T(n) =$~~ Time $= O(m \log^* n)$.

1.5. KKT Algorithm

KKT(G)

① run 3 rounds of Boruvka. $G=(V,E) \to G'=(V',E')$ with $n' \le n/8$, $m' \le m$

② if G' has 1 vertex, return

③ $E_1 \leftarrow$ random sample of E', each edge picked independently at probability 1/2.

④ $F_1 \leftarrow KKT(G_1 = (V', E_1))$

⑤ $E_2 \leftarrow$ all F_1-light edges in E'

⑥ $F_2 \leftarrow KKT(G_2 = (V', E_2))$

⑦ return F_2.

Complexity. $E[\#E_1] = \frac{1}{2} m'$, $E[\#E_2] = \frac{1}{2} n'$.

$T_{m,n} = \max_{\substack{G=(V,E),\\ |V|=n, |E|=m}} \{T_G\}$, $T_G \le C(m+n) + E[T_{G_1} + T_{G_2}]$

assume $T_{m,n} \le 2C(m+n)$

$T_G \le C(m+n) + E[2C(m_1+n')] + E[2C(m_2+n')]$

$\le C(m+n) + Cm' + 2Cn' + 4Cn' + 2Cn'$

$= C(m+m'+n+8n') \le 2C(m+n) = O(m+n)$.

Lecture 3

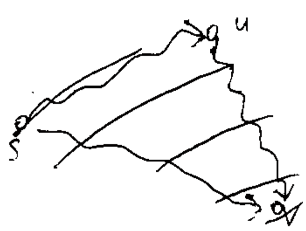
Sep. 19 2024

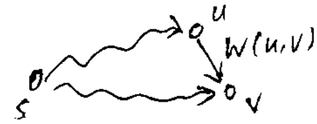
Shortest Paths

1. Single-Source Shortest Paths (SSSP)

1.1 Dijkstra's Algorithm. Omitted. 不能处理有负边之情形

<u>Definition 1</u> (Distance). Given a graph G and edge weights w, the ~~ma~~ minimum weight of any path from u to v is called "<u>distance</u>" $d_w(u,v)$

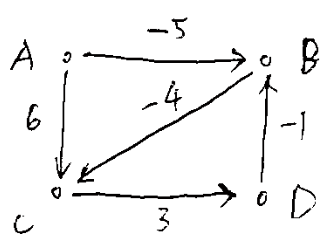


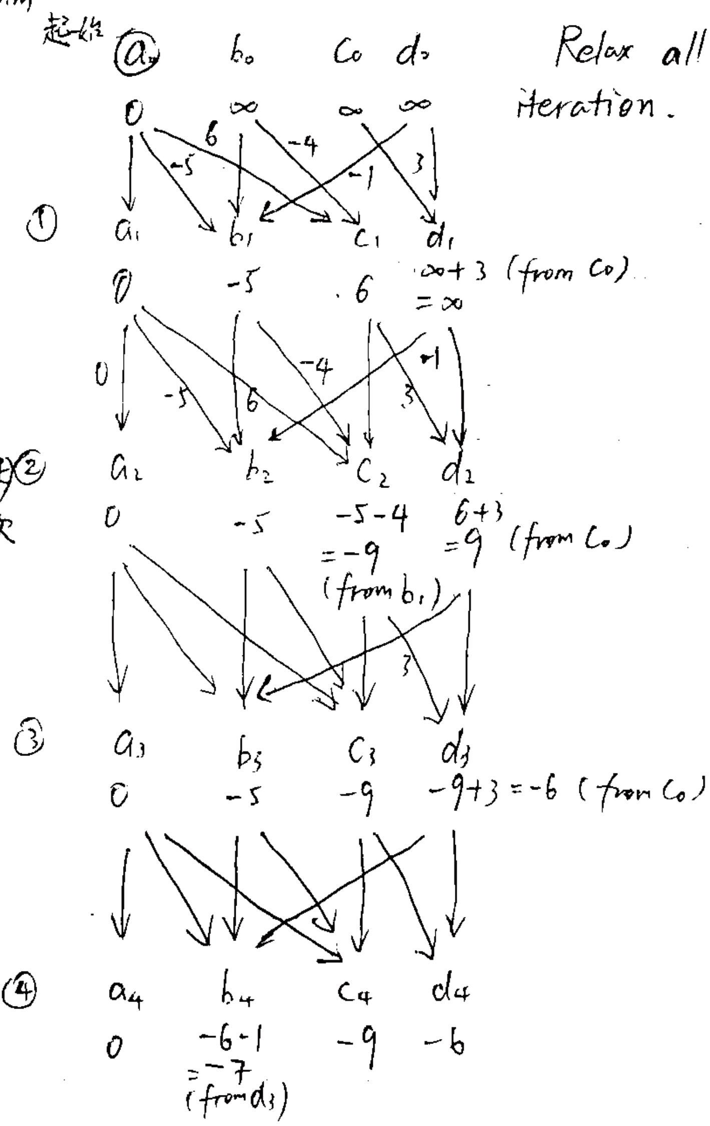


$d_w(s,v) \le d_w(s,u) + w(u,v)$

relax: $d_w(s,v) \leftarrow \min\{d_w(s,u)+w(u,v),\ d_w(s,v)\}$

1.2 Bellman-Ford Algorithm





Relax all d's in each iteration.

n个节点的图.
若 最短路至多(n−1)条边
(若无负环)(此时第n−1轮为结果)
若第n次迭代相较第n−1次
仍改变,则有负环.
并且,值改变的
节点是从起点出
发经由负环达到的,
距离 −∞.(这里的B)
且从这些节点又进一步
可达的节点,其距离亦
为 −∞(这里的C、D)

Bellman-Ford.

input: digraph $G=(V,E)$ with edge weights $w_e \in \mathbb{R}$ and src $s \in V$

output: shortest path from s to each ~~vertices~~ vertex or report a negative-weight cycle.

① $dist(S)=0$. $\forall v \in V-\{s\}$, $dist(v)=\infty$.

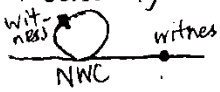
② for $|V|$ ~~times~~ iterations do

③ for $e=(u,v) \in E$ do

④ $dist(v) = \min\{dist(v), dist(u)+w(u,v)\}$

⑤ if any dist changes in the ~~last~~ (n-th) iteration, report a negative-weight cycle.

(Or, furthermore, we can label all vertices with their dist changed in the n-th iteration as "~~witte~~ witness", and all ~~witness~~ witness vertices together with all vertices reachable from them ~~also~~ (do a BFS) have dist $= -\infty$) Every negative-weight cycles have a witness, but not each witness is necessarily on a negative-weight cycle.



Complexity. $O(mn)$.

2. All-Pairs Shortest Paths (APSP)

2.1 Johnson's Algorithm.

Ideas behind Johnson's Algorithm:

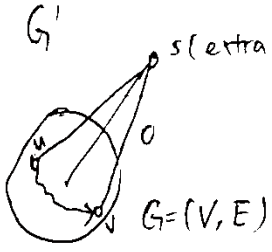
(1) re-weight the edges to have ≥ 0 weights & preserve shortest path (not dist)

(2) run n instances of Dijkstra's Algorithm.

Definition 2 (feasible potential). For a weighted digraph $G=(V,E)$, a function $\Phi: V \to \mathbb{R}$ is a feasible potential ~~of~~ if $\forall e=(u,v) \in E$,

$$\Phi(u) + w(u,v) - \Phi(v) \geq 0.$$

$$\Leftrightarrow \Phi(u) + w(u,v) \geq \Phi(v)$$



$w(s,v)=0$ for all $v \in V$. Run B-F. $\Phi(v) = d_w(s,v) \leq 0$.

If there is negative-weight cycle. ~~It~~ Johnson fails.

New weights of G': $\hat{w}(u,v) = \Phi(u) + w(u,v) - \Phi(v) \geq 0$

Shortest path: $f(u,v)$. $u \to k_1 \to k_2 \to \cdots \to v$

original weight: $w(u,k_1) + w(k_1,k_2) + w(k_2,k_3) + \cdots + w(k_x, v)$

new weight: $\Phi(u) + w(u,k_1) - \Phi(k_1)$

$+ \Phi(k_1) + w(k_1,k_2) - \Phi(k_2)$

$+ \cdots$

$+ \Phi(k_x) + w(k_x,v) - \Phi(v)$ ~~$= \Phi w(u,k_1) + w(u,k_1) + \cdots$~~

$= \underbrace{w(u,k_1) + w(k_1,k_2) + \cdots + w(k_x,v)}_{\text{original distance.}} + \underbrace{\Phi(u) - \Phi(v)}_{\text{constant for } u \text{ and } v.}$ 

⇙

Thus the original shortest path is preserved.

Now all weights are positive. Run $n \times$ Dijkstra's algorithm.

2.2 Floyd - Warshall Algorithm

Distance matrix.

F - W

initialization: $d(u,v) = \begin{cases} w(u,v), & \text{if } (u,v) \in E, \\ \infty, & \text{if } (u,v) \notin E \text{ and } u \neq v, \\ 0, & \text{if } u = v. \end{cases}$

input: A weighted digraph $G = (V, E)$

output: APSP for G.

① initialize $d(u,v)$ for all $u, v \in V$

② for all $t \in V$ do

③ for all $u, v \in V$. $u \neq v$. do // in fact. two layers of loops.

④ $d(u,v) \leftarrow \min\{d(u,v), d(u,t) + d(t,v)\}$.

Complexity. $O(n^3)$.

~~Min-Sum Product~~ ↓ Just like matrix multiplication.

$(A \odot B)_{ij} = \sum_{k=1}^{n} \min \; \min_k \{A_{ik} + B_{kj}\}$

$\underbrace{D \odot D \odot \cdots \odot D}_{k \text{ times}}$: minimum distance for any u, v in D with the path going through at most k edges.

Minimum distance / path:  $\bigodot_{i=1}^{n-1} D$.

Conjecture. $O(n^{3-\varepsilon})$, $\varepsilon>0$ APSP algorithm? Open problem.

Unit-weight, undirected graph.

Definition. (Adjacency matrix)

$$(A_G)_{ij} = \begin{cases} 1, & \text{if } (i,j)\in E, \\ 0, & \text{if } (i,j)\notin E. \end{cases} \qquad (A_{G^2})_{ij} = (A_G \times A_G)_{ij} > 0 \ \vee\ (A_G)_{ij} > 0.$$

Lemma 1. If $d_{x,y}$ and $D_{x,y}$ are shortest path distance between x, y in G and G^2 respectively, then $D_{x,y} = \lceil d_{x,y}/2 \rceil$.

Proof. In G,

(1) $x, a_1, b_1, a_2, b_2, \dots, a_k, b_k, y$. $\Rightarrow$ $2k+1$ edges. $d_{x,y} = 2k+1$.

~~$\Rightarrow$ in G^2, $x, b_1, b_2, \dots, b_k, y$. D_x~~

(2) $x, a_1, b_1, a_2, b_2, \dots, a_k, b_k, a_{k+1}, y$ $\Rightarrow$ $2k+2$ edges, $d_{x,y} = 2k+2$.

in G^2, $x, b_1, b_2, \dots, b_k, y$, $k+1$ edges. $D_{x,y} = k+1$

$k+1 = \lceil \frac{2k+1}{2} \rceil = \lceil \frac{2k+2}{2} \rceil$. $D_{x,y} = \lceil \frac{d_{x,y}}{2} \rceil$. □

Lemma 2. If $d_{x,y} = 2D_{x,y}$, then $\forall w \in N_G(y)$, $D_{x,w} \geq D_{x,y}$

Proof. Proof by contradiction. Suppose: $\exists\, w \in N_G(y)$, s.t. $D_{x,w} < D_{x,y}$

$\Rightarrow$ $D_{x,w} \leq D_{x,y} - 1$. $\Rightarrow$ $2D_{x,w} \leq 2D_{x,y} - 2$ $\Rightarrow$ $\boxed{2D_{x,w} < 2D_{x,y} - 1}$ (*)

Shortest path $x \rightsquigarrow w$ in $G + (w,y)$ forms a shortest path $x \rightsquigarrow y$.

$d_{x,w} + 1 \leq 2D_{x,w} + 1 \overset{(*)}{<} 2D_{x,y} = d_{x,y}$. a shorter path. Contradiction. □

Lemma 3. If $d_{x,y} = 2D_{x,y} - 1$, then $D_{x,w} \leq D_{x,y}$ $\forall w \in N_G(y)$. Moreover, $\exists\, z \in N_G(y)$, s.t. $D_{x,z} < D_{x,y}$

Proof. 

① For any $w \in N_G(y)$, consider shortest path $x \rightsquigarrow y$ in G, along with edge (y,w) $\Rightarrow$ $d_{x,w} \leq d_{x,y} + 1 = 2D_{x,y}$. By Lemma 1, $D_{x,w} = \lceil \frac{d_{x,w}}{2} \rceil \leq D_{x,y}$.

② Consider $z \in N_G(y)$ on a shortest path $x \rightsquigarrow y$. $d_{x,z} = d_{x,y} - 1 = 2D_{x,y} - 2$. $D_{x,z} = \lceil \frac{d_{x,z}}{2} \rceil < D_{x,y}$. □

Lemma 2,3 $\Rightarrow$ Corollary. $d_{x,y} = D_{x,y}$ iff. $\frac{\sum_{w \in N_G(y)} D_{x,w}}{\deg(y)} \geq D_{x,y}$.

(Because by Lemma 3 there is at least one such z s.t. "strictly less than").

Definition. (normalized adjacency matrix). The normalized adjacency matrix of G is $\hat{A}_{w,y} = \mathbb{1}_{(w,y) \in E} \cdot \frac{1}{\deg(y)}$. If D is the distance matrix of G^2, then $(D\hat{A})_{ij} = \sum_{w \in V} D_{x,w} \hat{A}_{w,y} = \frac{\sum_{w \in N_G(y)} D_{x,w}}{\deg(y)}$.

Distance matrix of $G = 2D - \mathbb{1}_{(D\hat{A} < D)}$

Seidel's: works for unit-weight undirected graphs.
input: $G = (V, E)$ with adjacency matrix A.
output: distance matrix of G.
① if $A = J$, or have recursed for $\log_2 n$ levels, return A.
② else
③ $A' \leftarrow (A * A) \vee (A)$
④ $D \leftarrow \text{Seidel}(A')$ // recursion
⑤ return $2D - \mathbb{1}_{(D\hat{A} < D)}$.

Sep. 26 2024

Lecture 4

Divide and Conquer Algorithms

1. FFT

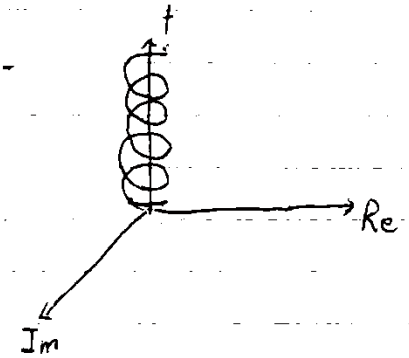
Fourier Series.

任意周期(非周期)时函数可以表示为正弦/余弦函数(或复指数函数)的无穷级数.

$$f(t)=\frac{a_0}{2}+\sum_{n=1}^{+\infty}\left(\underbrace{a_n}_{\text{相位}}\cos(nw_0t)+\underbrace{b_n}_{\text{振幅}}\sin(nw_0t)\right),\quad w_0=2\pi/T \text{ 基频}$$

or.

$$f(t)=\sum_{n=-\infty}^{+\infty}c_n e^{i\cdot nw_0t},\quad c_n=\begin{cases}\frac{1}{2}(a_n-ib_n), & n>0,\\ a_n/2, & n=0,\\ \frac{1}{2}(a_{-n}+ib_{-n}), & n<0.\end{cases}$$



$$\langle f,g\rangle=\int_a^b f(t)g(t)\,dt.$$

~~For $n_1\neq n_2$~~ $\quad f(t)=e^{in_1w_0t},\ g(t)=e^{in_2w_0t}.$

$$\frac{1}{2\pi}\langle f,g\rangle=\frac{1}{2\pi}\int_{-\pi}^{\pi}e^{in_1w_0t}e^{in_2w_0t}\,dt$$

$$=\begin{cases}0, & n_1+n_2\neq 0,\\ 1, & n_1+n_2=0.\end{cases}\quad \text{正交性}$$

任意二维空间之函数可表示为无穷维空间上集合.

求 c_n 就是求 $f(t)$ 在所有坐标轴之投影(内积),即

$$c_n=\frac{1}{T}\int_{-\pi/2}^{\pi/2}f(t)\,e^{-inw_0t}\,dt$$

(对周期为 T 之函数).

对非周期函数. $T\to\infty$, $w_0=2\pi/T\to 0$, $nw=w_0$ 任意实数

$$F(w)=\int_{-\infty}^{+\infty}f(t)e^{-iwt}\,dt$$

Discrete Fourier Transformation. (DFT)

In reality, we ~~don't~~ often do not have the function $f(t)$, but only samplings of a signal periodically, every t time, i.e., discrete signals along the time axis.

$$Y[k]=\sum_{n=0}^{N-1}X[n]e^{-\frac{2\pi}{N}kni},\quad k=0,1,\dots,N-1,\quad T=N.$$

卷积定理

(1) 时域卷积 = 频域乘积　$f(t) \otimes g(t) = F(w) \odot G(w)$

(2) 时域乘积 = 频域卷积.　$f(t) \odot g(t) = F(w) \otimes G(w)$

⊗：卷积

⊙：点对相乘，等价不完全相同.

Proof of (1)　$(f \otimes g)(t) = \int_{-\infty}^{+\infty} f(\tau) g(t-\tau) d\tau$

$F((f \otimes g)(t)) = \int_{-\infty}^{+\infty} \left(\int_{-\infty}^{+\infty} f(\tau) g(t-\tau) d\tau \right) e^{iwt} dt.$

交换积分顺序.　$= \int_{-\infty}^{+\infty} f(\tau) \left(\int_{-\infty}^{+\infty} g(t-\tau) e^{-iwt} dt \right) d\tau.$

令　$u = t-\tau \Rightarrow dt = du.$

$= \int_{-\infty}^{+\infty} f(\tau) \left(\int_{-\infty}^{+\infty} g(u) e^{-iw(u+\tau)} du \right) d\tau = \int_{-\infty}^{+\infty} f(\tau) e^{-iwt} d\tau \int_{-\infty}^{+\infty} g(u) e^{-iwu} du$

$= F(w) \odot G(w)$　□

$$Y[n] = \sum_{t=0}^{N-1} X[t] \, e^{-\frac{2\pi}{N} nti} \qquad 令\ w_N = e^{\frac{2\pi}{N} \cdot i}$$

$$= \sum_{t=0}^{N-1} X[t] \, w_N^{-tn}$$

$$= \sum_{\substack{t=0, \\ t\ even}}^{N-1} X[t] \, w_N^{-tn} + \sum_{\substack{t=1, \\ t\ odd}}^{N-1} X[t] \, w_N^{-tn}$$

$$= \sum_{t=0}^{N/2-1} X[2t] \, w_N^{-n \cdot 2t} + \sum_{t=0}^{N/2-1} X[2t+1] \, w_N^{-n(2t+1)}$$

周期性 ↖

$$= \underbrace{\sum_{t=0}^{N/2-1} X[2t] \, w_{N/2}^{-t(n \% \frac{N}{2})}}_{\text{Size } N/2 \text{ DFT}} + w_N^{-n} \underbrace{\sum_{t=0}^{N/2-1} X[2t+1] \, w_{N/2}^{-t(n \% \frac{N}{2})}}_{\text{Size } N/2 \text{ DFT}}.$$

```
FFT(X[0, 1, ..., N-1], w_N, N)
if N=1 return X[0]
Y_even ← FFT(X[0, 2, ..., N-2], w_{N/2}, N/2)
Y_odd  ← FFT(X[1, 3, ..., N-1], w_{N/2}, N/2)
for n = 0 to N-2 do
    Y[n] = Y_even[n % N/2] + w_N^{-n} Y_odd[n % N/2]
return Y
```

$T(N) = 2T(N/2) + O(N).$

$T(N) = O(N \log N)$

数据访问复杂度. # of cache miss : read : $2\frac{N/2}{B/2} + 4\frac{N/4}{B/4} + \cdots + 2^x\frac{N/2^x}{B/2^x} + \underbrace{2^{\log_2 B}\cdot\frac{N}{1} + \cdots + N}_{\log_2\frac{N}{B}\text{ times}}$

cache size M

cache line = B elems

memory ∞

Yodd / Yeven takes elements from cacheline every other one.

$= O(N \log N)$

Lower bound of # of cache miss: $\frac{N\log N}{M\log M}\cdot\frac{M}{B} = \Omega\left(\frac{N}{B}\log_M N\right)$

Cache_Oblivious_FFT ($X[0,1,\ldots,N-1]$, ω_N , N)

if $N = O(1)$, compute DFT directly.

else for any factorization $N = N_1 \times N_2$

// regard $X: N_1 \times N_2$, $Y: N_2 \times N_1$ matrices.

$Y[i_2 \times N_1 + i_1] = \sum_{j=0}^{N-1} X[j_1 \times N_2 + j_2]\, \omega_N^{-\frac{(i_2 N_1 + i_1)(j_1 N_2 + j_2)}{n}}$

$\hookrightarrow = \omega_{N_2}^{-i_2 j_2}\, \omega_{N_1}^{i_1 j_1}\, \omega_N^{i_1 j_2}$

$= \sum_{j_2=0}^{N_2-1} \left(\left(\sum_{j_1=0}^{N-1} X[j_1 N_2 + j_2]\, \omega_N^{-i_1 j_1} \right) \omega_N^{-i_1 j_2} \right) \cdot \omega_{N_2}^{-i_2 j_2}$

matrix transpose

$T(N) = N_2 T(N_1) + N_1 T(N_2) + O(N/B) = O(N \log N)$, $N_1 = N_2 = N^{1/2}$

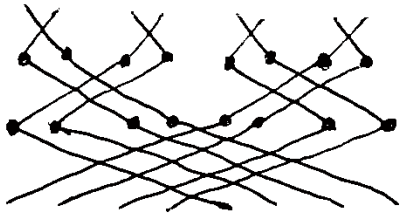
Data access complexity

$$Q(N) \le \begin{cases} O(1 + N/B) & , N \le \varepsilon M \ , 0 < \varepsilon < 1 \\ N_1 Q(N_2) + N_2 Q(N_1) + O(1 + N/B) = O\left(\frac{N}{B}\log_M N\right) & , N > \varepsilon M \end{cases}$$

↳ LB. reached.

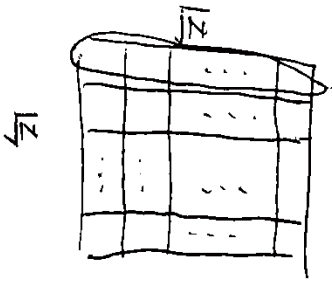
Further optimization:

bit-reversal permutation $\overrightarrow{0001} \to \overleftarrow{1000}$, $\overrightarrow{0101} \to \overleftarrow{1010}$.



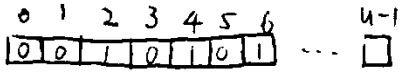
matrix

No ∧ transpose overhead. $O\left(\frac{N}{B}\log_{M/B} N\right)$



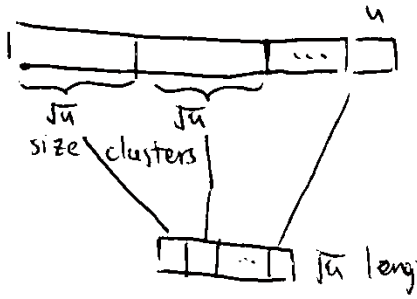
2. van Emde Boas (vEB) tree

Data structure. n elements $\in \{0, 1, \ldots, u-1\}$.
Insert, Delete, Successor $O(\log \log u)$ time. u: size of domain of n

① bit vector 

$V[i] = \begin{cases} 0, & \text{if } i \text{ is not an element,} \\ 1, & \text{if } i \text{ is an element.} \end{cases}$

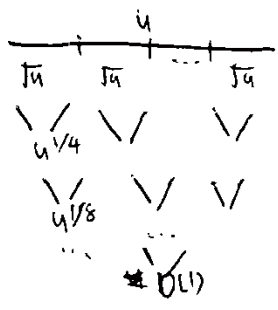
insert / delete: $O(1)$.
successor: $O(u)$.

② 

Summary [i] = 1 if there is some 1 in the i-th cluster.

~~insert/ delete: O(1)~~ insert: $O(1)$, delete: $O(\sqrt{u})$.
↳ whether summary [..] should change to 0 depends on whether the current cluster still contains any 1. Have to scan the cluster.

successor: $O(\sqrt{u})$. (at most $3 \times \sqrt{u}$)

③ 

insert: $T(u) = 2T(\sqrt{u}) + O(1)$
$= O(\log u)$
successor: $T(u) = 3T(\sqrt{u}) + O(1)$
$= O(\log u)$

④ 

minimum and maximum value stored in the substructure.

delete/ insert: $T(u) = 2T(\sqrt{u}) + O(1)$
successor(V, x) : $T(u) = T(\sqrt{u}) + O(1)$
↳ the structure

if $x < V.min$
$V.min = x$
if $x > V.max$
$V.max = x$.

Oct 10 2024 Lecture 5

Dynamic Programming (DP)

1. SRTBOT Model.

SRTBOT: Subproblem, Relate subproblems, Topological order, Base cases, Original, Time analysis (一般不能用递归表达式求解)

$Fib(n) = Fib(n-1) + Fib(n-2)$.

Subproblem: $Fib(i)$

Relate: $Fib(i) = Fib(i-1) + Fib(i-2)$

Topo-order: increasing i. $0, 1, 2, 3, \dots$

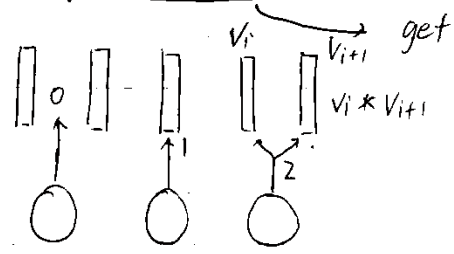
Base: $F(0)=0$, $F(1)=1$

Original: $F(n)$

Time: Fib 指数增长. 但表示 字数 $O(\frac{\log a^n}{W})$ → 位数 / 字长 $= O(\frac{n}{W})$ 加法开销

故 求 n 项需 $n \cdot O(\frac{n}{W}) = O(n^2/W)$.

Bowling. n pins, each with value v_i (can be <0). Balls can hit 0 pin, 1 pin or 2 (adjacent) pins to maximize total value (sum). get the point = product of their values.



Subproblem. $B(i,j)$ pins $i, i+1, \dots, j-1$.

Relate. $m = \lfloor \frac{i+j}{2} \rfloor$, $B(i,j) = \max \{ v_m v_{m+1} + B(i,m) + B(m+2,j)$
$, B(i,m+1) + B(m+1,j) \}$

Topo-order. increasing $(j-i)$

Base: $B(i,i)=0$, $B(i,i+1) = \max\{0, v_i\}$

Original. $B(0,n)$

Time. There are n^2 possible (i,j) pairs. Each 'Relate' step takes $O(1)$ time.
$\Rightarrow O(n^2)$ time.

Unefficiency: $\max \{ v_m v_{m+1} + B(i,m) + B(m+2,j)$
$, B(i,m+1) + B(m+1,j) \}$

Conceptually, $B(i,m)$ and $B(i,m+1)$ are related. $B(i,m+1)$ "contains" $B(i,m)$.

Subproblem. $B(i)$: $i, i+1, \ldots, n-1$
Relate. $B(i) = \max\{B(i+1),\ v_i + B(i+1),\ v_i v_{i+1} B(i+2)\}$
Topo-order. decreasing i.
Base. $B(n) = 0$
Original. $B(0)$
Time. $O(n) \cdot O(1) = O(n)$.

Longest Increasing Sequence (LIS)
- Given a string A, find a longest (not necessarily contiguous) subsequence of A that strictly increases (lexicographically).
- Example. $A =$ "carbonhydrate", sol = "abort".

Subproblem. $X(i)$. LIS of A including $A[i]$.
Relate. $X(i) = \max\{1 + X(j) \mid i<j<|A|,\ A[j]>A[i]\} \cup \{1\}$. → Brute-Force searching all characters after $A[i]$. $O(n)$
Topo-order. decreasing i
Base.
Original. $\max\{X(i) \mid 0 \le i < |A|\}$
Time. $O(n) \cdot O(n) = O(n^2)$.
- $O(n)$: Brute-Force search
- $O(n)$: # of subproblems.

Alternating Coin Game.
- Given a sequence of n coins of values $\{v_0, v_1, \ldots v_{n-1}\}$. $v_i > 0$.
- Two players "you" and "me". Each round, pick the first or the last coin.
- Goal: maximize the total value of "me" ~~"you"~~

Subproblem. $X(i,j)$ max val "me" can get from # i to j. $0 \le i \le j \le n$.
Relate. ~~max~~ $X(i,j) = \max\{v_i + \underline{(\sum_{k=i+1}^{j} v_k) - X(i+1,j)},\ v_j + (\sum_{k=i}^{j-1} v_k) - X(i,j-1)\}$
↘ if computed with prefix sum, $O(1)$, but sometimes we do not have "subtraction". e.g. ~~mean of numbers~~. "+" := mean of total value taken.
Topo-order. increasing $(j-i)$
Base. $X(i,i) = v_i$
Original. $X(0, n-1)$
Time. $O(n^2) \cdot O(n) = O(n^3)$.
- $O(n^2)$: # of (i,j) pairs.
- $O(n)$: cost of sum.

Improving. "me" 能拿到的最大值, 对 "you" 希望 "me" 拿到的最小值.

~~Sup~~ Subproblem. $X(i, j, p)$. ~~max val~~ $p \in$ {"me", "you"}. ~~can get from i to j, 0≤i≤j≤n~~

Relate. $X(i, j, \text{"me"}) = \max\{V_i + X(i+1, j, \text{"you"}), V_j + X(i, j-1, \text{"you"})\}$.

$X(i, j, \text{"you"}) = \min\{X(i+1, j, \text{"me"}), X(i, j-1, \text{"me"})\}$.

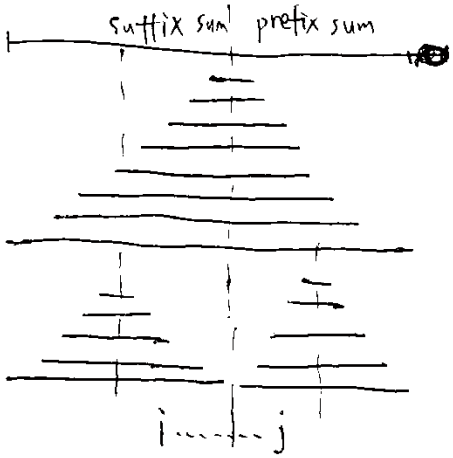
Topo-order. increasing $(j-i)$.

Base. $X(i, i, \text{"me"}) = V_i, X(i, i, \text{"you"}) = 0$ (只有一个, "you" 一定会取走).

Original. $X(0, n-1, \text{"me"})$.

Time. $O(n^2) \cdot O(1) = O(n^2)$.

Sum from i to j th, without "subtraction" operation:

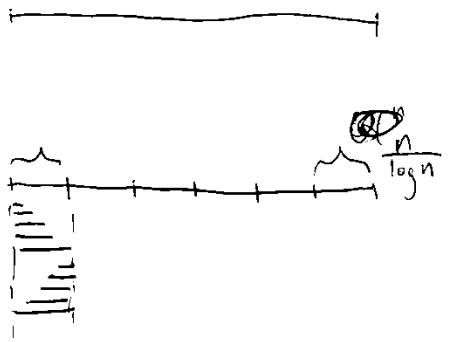


$T_1(n) = O(n) + 2T_1(n/2)$.

$= O(n \log n)$.

$O(1)$.

Optimize to $\alpha(n)$:



$T_2(n) = O(n) + T_1(\frac{n}{\log n}) + \frac{n}{\log n} T_2(\log n)$

$= O(n) + \frac{n}{\log n} T_2(\log n)$

$\frac{n}{\log n} \log n + \frac{\log n}{\log \log n} T_2(\log \log n)$

...

$= O(n \log^* n)$.

$\frac{n}{\log^* n} \Rightarrow O(n \log^{**} n), \ldots, O(n \alpha(n))$

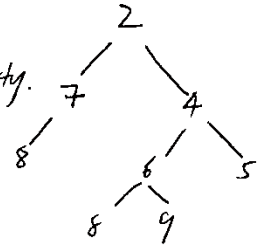
LCA & RMQ

RMQ (random minimum query). Special case (+ := min(_,_)) of partial sum.

Cartesian tree , 1984 STOC .

A.

8,7,2,8,6,9,4,5 ⟹



root: min of A. A[i] min-heap property.

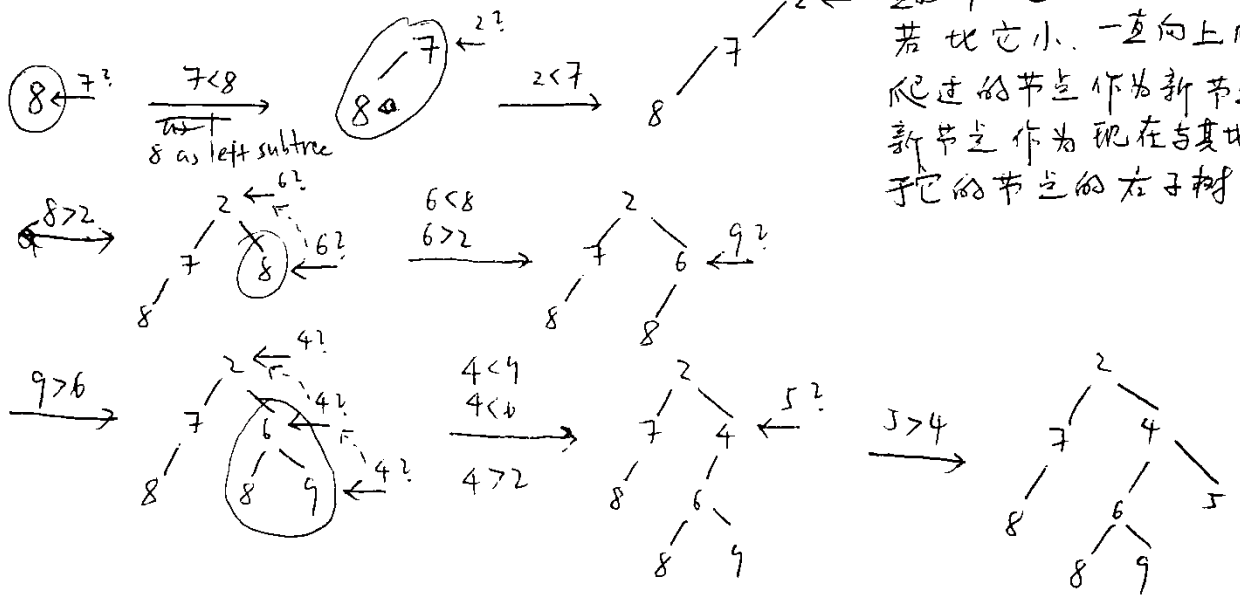
left subtree: cartesian tree of A[<i]

right subtree: cartesian tree of A[>i].

Get Cartesian tree in O(n) time:

每次从树最右例插. 若要插的新节点比它大, 直接作为它的右子树

若比它小, 一直向上爬直到大. 爬过的节点作为新节点之左子树. 新节点作为现在与其比较而大于它的节点的右子树.

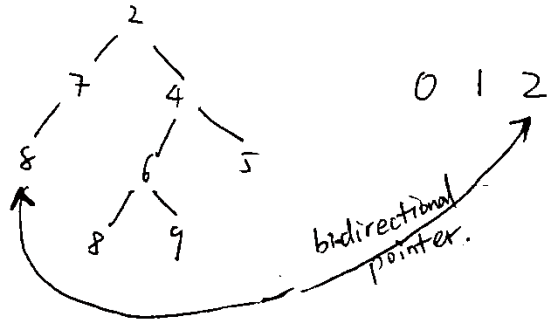


LCA ⇔ RMQ.

RMQ → Cartesian tree → LCA in Cartesian tree is the minimum we expect.

LCA → in-order traversal → RMQ.

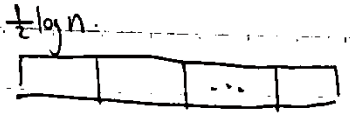
~~LCA~~ RMQ → LCA → $^{\pm 1}$RMQ.



LCA(i,j)
= RMQ(first(i), first(j)).

No.

Date



$2^{\frac{1}{2}\lg n} = \sqrt{n}$. $O(\sqrt{n}\lg^2 n)$ $\log \frac{1}{2}\lg n = O(n)$ bits

Lecture 6

Oct. 17 2024

Amortized Algorithms

1. Intro.

key observation. the only way to produce an expensive operation is to "set it up" with a large number of cheap operations beforehand.

2. Dynamic Increasing Array.

1) Init. $n=0$, $c=1$.
2) Append(x).
```
if c = n
    grow()
write x at position n.
increment n.
```
3) get(i). get i-th element of array
4) grow()
```
double c.
create a new array of 2c
move elements from old to new array
```

2.1 Aggregate Method

Lemma 1. The cost of a sequence of m append operations is $\leq 3m$.

Proof. ① No grow. cost m

② Some grow happens. Suppose after m appends, $c = \lceil\lceil m \rceil\rceil$ (smallest power of 2 that is $\geq m$).

1) $m \leq \lceil\lceil m \rceil\rceil \leq 2m$. ~~(writes)~~

2) Current capacity $=c$, latest grow: $c/2 \to c$. cost $= c/2$.

total cost of grow: $1 + 2 + 4 + \cdots + c/2 = c/2 = \lceil\lceil m \rceil\rceil / 2 \leq m$

$\leq 1+2+4+\cdots+m \leq 2m$.

$\Rightarrow$ m writes + $2m$ copies (due to grows) $= 3m$.

2.2 Banker's Method (零存整取)

amortized cost = actual cost + credit it pays − credits it consumes.

Lemma 2. The amortized cost (using banker's method) of `Append` is 3.

Proof. For every append, (1) actual cost = 1, + pay extra cost = 2 = 3.
(2) grow occurs. $c = n$ $n/2 \cdot 2 = n$.

$1 + n + 2 - n/2 \times 2 = 3$

1: 写一个 ↓ | n: move ↓ | +2: 自己存2个

它们存的之前grow时已用过 | 它们存的 credits 还没用过.

2.3 Potential Method.

State of the whole data structure (at some point i) $S_i \mapsto \Phi(S_i) \in \mathbb{R}$. Φ s.t. $\forall i, \Phi(S_i) \geq \Phi(S_0)$

Amortized cost of the i-th operation $ac_i = c_i + \Phi(S_i) - \Phi(S_{i-1})$

c_i: actual cost

使数据结构之状态从 S_{i-1} 变为 S_i

$\sum_{i=1}^{n} ac_i = \sum_{i=1}^{n} c_i + \Phi(S_n) - \Phi(S_0)$

$\sum_{i=1}^{n} ac_i$: 均摊开销之累加 (实际开销之上界)

$\geq$

$\sum_{i=1}^{n} c_i$: 实际开销

$\Phi(S_n) - \Phi(S_0) \geq 0$ by definition of potential functions.

使宜操作每次让势能增加一点, 贵的操作让势能大幅下降.

Lemma 3. ~~Append~~ $\Phi(n, c) = 2(n - c/2)$ $n \geq c/2 \Rightarrow \Phi \geq 0$, $\Phi(S_0) = \Phi(0, 1) = 0$.

Proof. Append. $c_i = 1 + \Delta\Phi_i = 3 = O(1)$

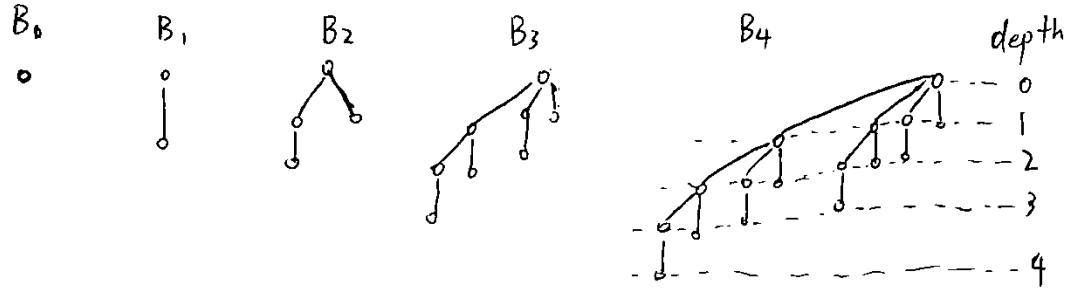
grow. $c_i = n$, $\Delta\Phi_i = 2(n - 2n/2) - 2(n - n/2) = -n$. $c_i + \Delta\Phi_i = 0$. $ac_i = 0$.

3. Binomial Heap. (not binary heap)

insert	ins(H, x)	O(1) amortized	O(log n)
minimum	min(H)	O(1)	O(1)
extract_min(H)		O(log n)	O(log n)
union(H1, H2)		O(1)	O(n)
decrease_key(H, x, k)		O(log n)	O(log n)
delete(H, x)		O(log n)	O(log n)

} Fib heap amortized O(1).

Binomial Tree. B_0 : binomial tree with 1 node. $\forall k>0$, $B_k = 2B_{k-1}$, one with smaller key as the root of another



Some useful properties of B_k.

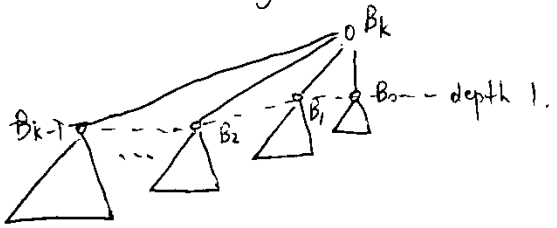
(1). it has 2^k nodes.

(2) its height is k

(3) there are exactly $\binom{k}{i}$ nodes at depth i

(4) root has degree (rank) k.

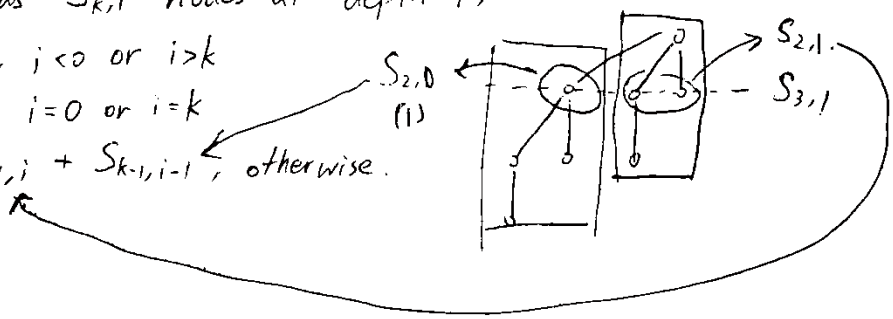
(5)



(1). depth $0 = i$. $\binom{k}{0} = 1$.

Suppose B_k has $S_{k,i}$ nodes at depth i,

$$S_{k,i} = \begin{cases} 0, & i<0 \text{ or } i>k \\ 1, & i=0 \text{ or } i=k \\ S_{k-1,i} + S_{k-1,i-1}, & \text{otherwise.} \end{cases}$$



$$\Rightarrow S_{k,i} = \mathbb{1}[k \geq i \geq 0]\left(S_{k-1,i} + S_{k-1,i-1} + \mathbb{1}[i=k=0]\right) \quad (2).$$

$$S_k(x) = S_{k,0} + S_{k,1}x + S_{k,2}x^2 + \cdots + S_{k,k}x^k = \sum_{i=0}^{k} S_{k,i}x^i$$

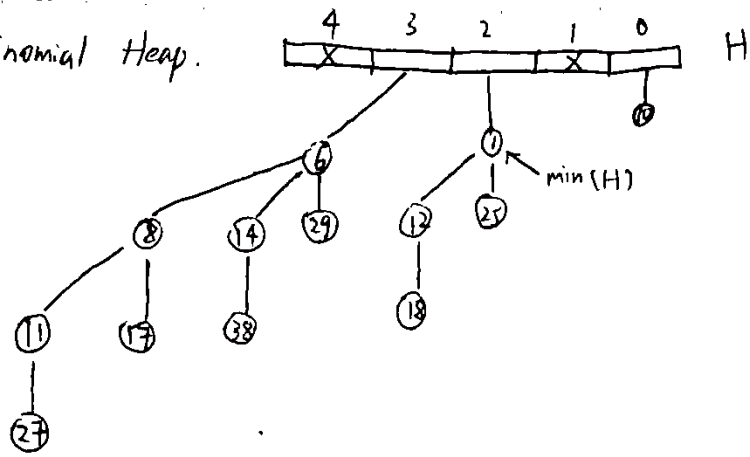
$$= \sum_{i=0}^{k} S_{k-1,i}x^i + \sum_{i=0}^{k} S_{k-1,i-1}x^i + \mathbb{1}[k=0]$$

$$= \sum_{i=0}^{k-1} S_{k-1,i}x^i + x\sum_{i=0}^{k-1} S_{k-1,i}x^i + \mathbb{1}[k=0]$$

$$= S_{k-1}(x) + x\,S_{k-1}(x) + \mathbb{1}[k=0]$$

$$S_k(x) = (1+x)S_{k-1}(x) + \mathbb{1}[k=0] \Rightarrow S_k(x) = (1+x)^k. \Rightarrow S_{k,i} = \text{系数}\binom{k}{i}.$$

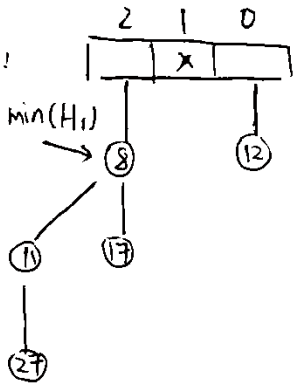
Binomial Heap.



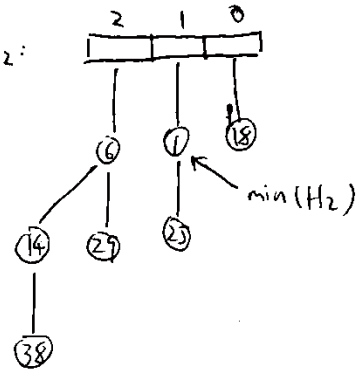
B_k

rank(t) = rank(root of t) = k.

H_1:

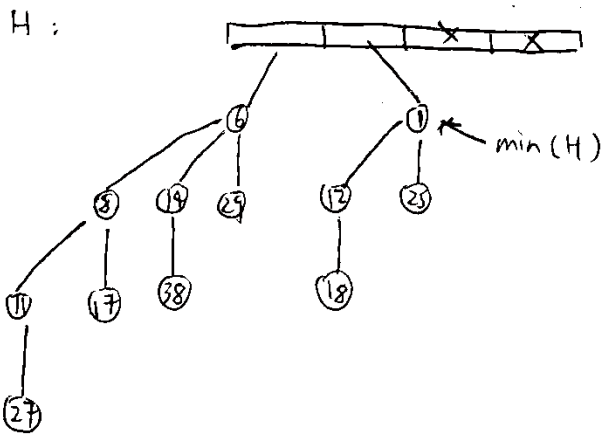


H_2:



合并: $\lfloor 3\log_2 3 \rfloor + 1 = 4$. 从低位向高位. 无树则仍 X, 仅一个则取那个. 都有则合并. 向上进一位, 原该位设为 X. 进位又遇两棵则加起来再进一位. 又遇 2 棵则任取一棵加起, 和进一位. 另一棵留原位. 正如二进制加法.

H:



Ins (H, x).

(1) $H' \leftarrow$ make_heap (x)

(2) union (H, H'), $O(\log n)$ worst_case extract_min.

extract_min (H)

(1) 删除 min element

(2) 将其子树结点直接构建一个新堆 H'.

(3) union (H, H')

Complexity: $\phi(S_i) = c \times$ # of trees in the binomial heap. c, constant to be decided.

$\phi(S_0) = 0. \Rightarrow \phi(S_i) - \phi(S_0) \geq 0.$

make_heap (x). $c_i = 1$, $\Delta\phi_i = c \cdot 1 = c \Rightarrow ac_i = c_i + \Delta\phi_i = O(1)$

ins (H, x). $c_i = 1 + k$, k: (k-1) carryings occurr, k positions scanned.

$\Delta\phi_i = c(1-(k-1))$ ↘ each carrying, decrease one tree.

$ac_i = c_i + \Delta\phi_i = 1+k+c-c(k-1) = 2+c+(1-c)(k-1) \leq 2+c. = O(1).$

$k \geq 1$ (at least one position scanned). take $c > 1$. ins: O(1)

union (H_1, H_2) $c_i = k$ k positions scanned

$\Delta\phi_i = -c \cdot l$. l: l links occur

$ac_i = c_i + \Delta\phi_i = k - cl = O(\log n).$

extract_min (H). $O(\log n)$, r: 原堆中指向 min(H). 留作习题.

O(1) union.

Oct. 24 2024 Lecture 7

P/NP

1. DP

(1) Rod Cutting. Given a rod of length L and value $V(l)$ of ~~length~~ rod of length l for all $l \in \{1, 2, \dots, L\}$,

Goal: cut the rod to maximize total value of cutted rod pieces.

Example. $L = 7$. $V = [0, 1, 10, 13, 18, 20, 31, 32]$ (l: 0 1 2 3 4 5 6 7)

Greedy. always maximize the current $V(l)/l$. (value per unit length)

$V(l)/l = [1, 5, 13/3, 18/4, 4, 31/6, 32/7]$

in this way we get $[6, 1]$, $V = 31 + 1 = 32$.

However, $[2, 2, 3]$, $V = 10 + 10 + 13 = 33$. $[6, 1]$ not optimal.

DP. Subproblem. $X(l)$. 不管怎么切割，能从长度为 l 的棒得到的最大价值

Relate. $X(l) = \max\{V(p) + X(l-p) \mid p = 1, 2, \dots, l\}$

Topological. increasing l.

Base. $X(0) = 0$

Original. $X(L)$.

Time. $L \cdot L = O(L^2)$

input. $(L+1)+1$ integers. (输入问题规模). rod cutting is polynomial.

(2). Subset Sum. Input: a sequence of n positive integers $A = \{a_0, a_1, \dots, a_{n-1}\}$, $T \in \mathbb{N}_+$.

Goal. Is there a subset of A that sums exactly to T, i.e, $\exists A' \subset A$, $\sum_{a \in A'} a = T$.

Example. $A = \{1, 3, 4, 12, 19, 21, 22\}$, $T = 47$. result: "yes".

Subproblem. $X(i, t)$: $A[i..]$, t, exist such subset of $A[i..]$ summing up to t?

Relate. $X(i, t) = X(i+1, t - A[i])$ or $X(i+1, t)$

Topological. decreasing i

Base. $X(i, 0) = Y$, $X(n, t) = N$ if $t > 0$.

Origin. $X(0, T)$.

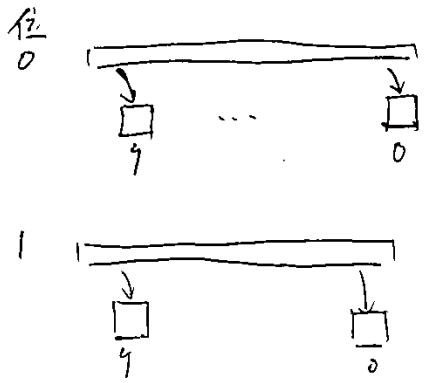
Time. $n \cdot T \cdot O(1) = O(nT)$

(# of i's. # of possible t's.)

size of input: $n+1$ (A) $\nearrow$ $T \in \mathbb{N}_+$ $\nearrow$. If $T = e^n$, $O(nT)$ is not a polynomial with respect to the input size n. 「pseudo-polynomial」.

(3) Radix Sort. radix $= r$. n integers. $\rightarrow$ $(a, b, c, \ldots)_r$ 「按 r 进制写」
msb lsb

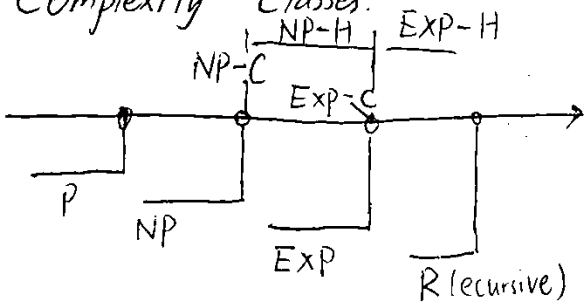
$(1,0,2,3)_{10}$



按最 lsb 排序
↓
按收序拿出
↓
按第 2 低位排序
↓
⋮

time: $O(n \log_n^c U)$. weakly polynomial

2. Complexity Classes.



Proving NP-Completeness / hardness.
reduction. $A \rightarrow B$. reduce every instance of A to some instances of B. (in poly time)
$\Rightarrow$ A 的难度 $\leq$ B 的难度.

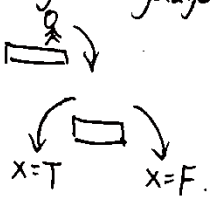
$A \rightarrow B$, then
(1) $B \in P \Rightarrow A \in P$
(2) $A \in$ NP-C/H $\Rightarrow B \in$ NP-C/H.

3. Several Classic NP Problems.
3SAT. Given a boolean formula of form $(x_1 \vee x_3 \vee \bar{x}_6) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_7) \wedge \cdots \wedge$ ___
(It can be proven that any boolean expression can be written in 3SAT form)
Question. Is there any assignment of vars to T/F, s.t. entire formula evaluates to T?
$3SAT \in$ NP-C.
Proof idea. (1) $3SAT \in NP$.
(2). 只能从定义: 比任何 NP 问题更难.
任何 NP 问题都能用仅包含 $\wedge$, $\vee$, $\neg$ 的电路实现. (所有 NP 问题归约到只包含 $\vee$, $\wedge$, $\neg$ 的表达式)

Super Mario: 3SAT → SM. 对于{任何 3SAT 实例，构造 SM 之一关}.

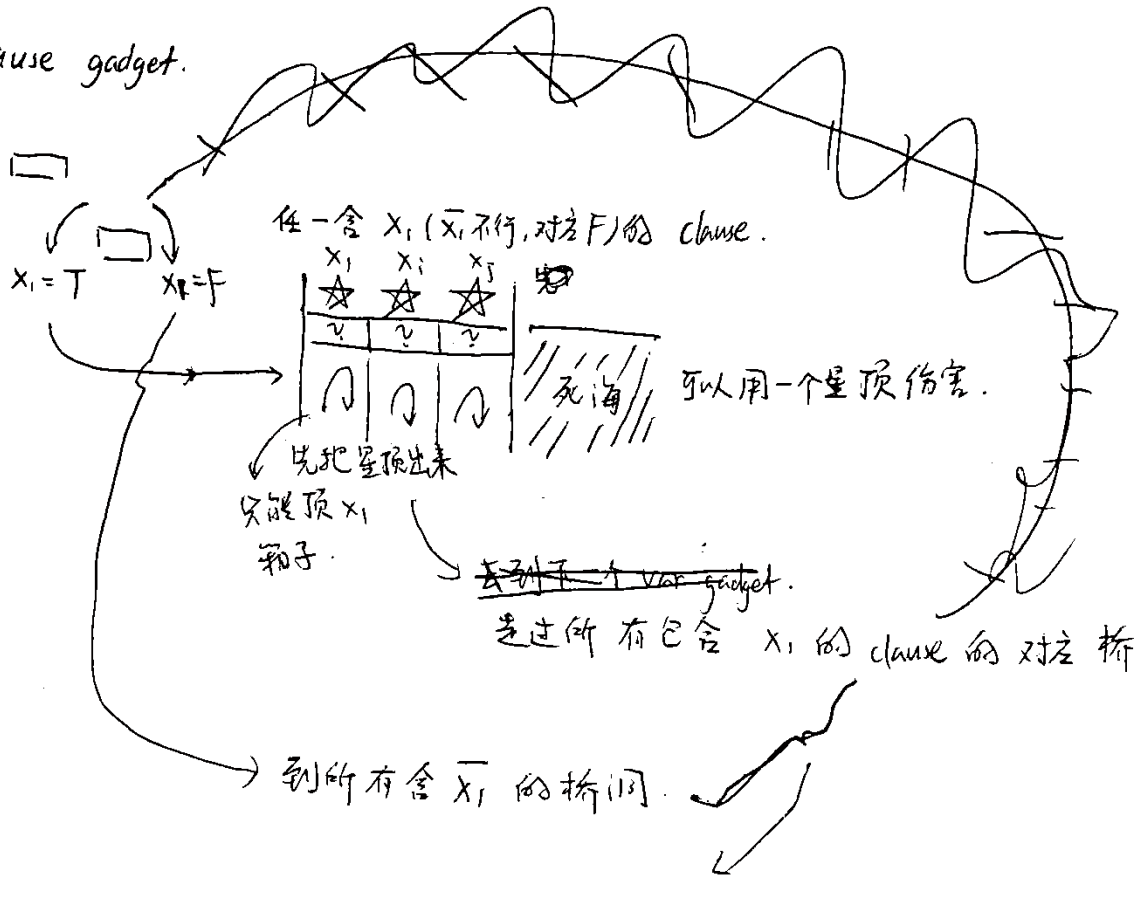
n variables, m clauses.

var ~~gadget~~ gadget.



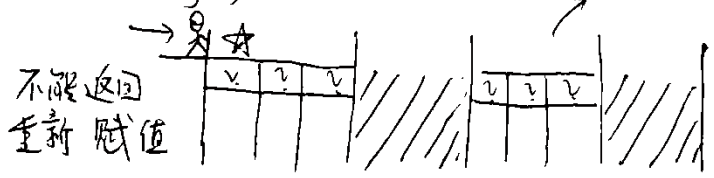
每个变量相应构造一个 var gadget.

clause gadget.



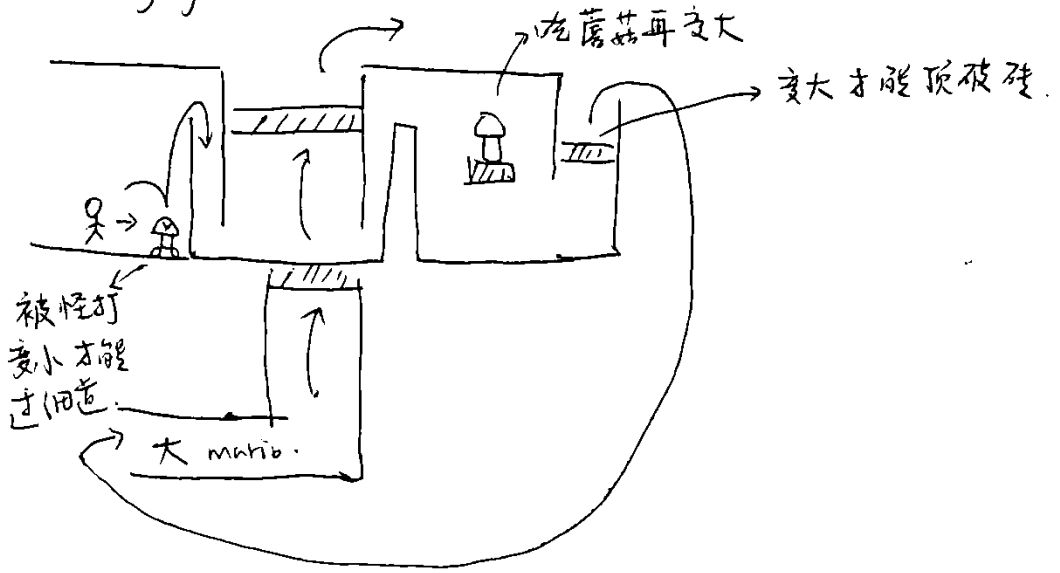
走完所有 x_i 的 var gadget. 顶完所有星.

每个变量间
Crossover gadget 连接.



若有一处无星. 则该~~赋值~~ clause 未被该赋值满足. 过不去.

Crossover gadget:



3DM (generalization of classical 2-sex marriage problem).

2-sex marriage problem $\in P$.
n male, n female. 每个男性有若干候选(结婚对象)女性; vice versa.
是否可以找到 这样的匹配: n个 (男,女) 对. 且互相都是候选对象.

3DM: 3-sex. ~~n × sex$~~ $n \times X$, $n \times Y$, $n \times Z$.

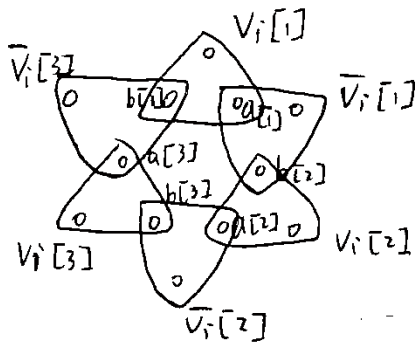
$T \subset X \times Y \times Z$. 可接受的结婚组合.

是否可以找到 n个 3-匹配, 是 X-Y-Z 的, 且每个 x、y、z 在其中出现一次且仅一次.

NP-H by reduction from 3SAT.

$V = \{v_1, v_2, \ldots, v_n\}$, $C = \{c_1, c_2, \ldots, c_m\}$.

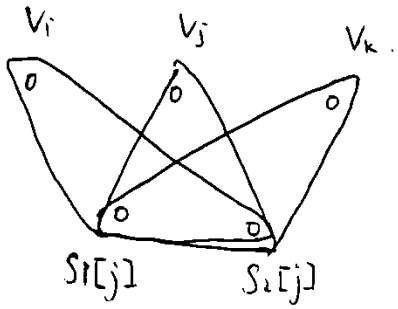
(1) variable gadget.



2m个 三角.

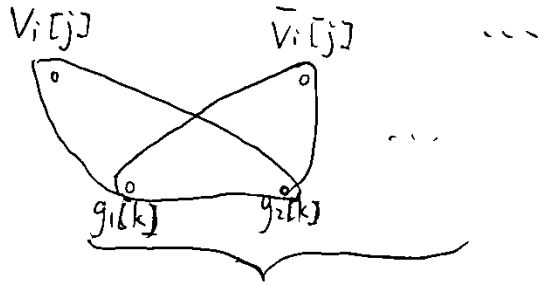
v_i 选 true, 选择所有 $\bar{v}_i[\cdot]$ 覆盖内部 a、b 点. 所有 $v_i[\cdot]$ 可以出去对 clause gadget 赋值.

若有 clause: $V_i \vee V_j \vee V_k$.



选择一个 $V_{i,j,k}$ 覆盖 S_1, S_2.
但不能两个. 否则 婚姻不稳定.

garbage collector gadget



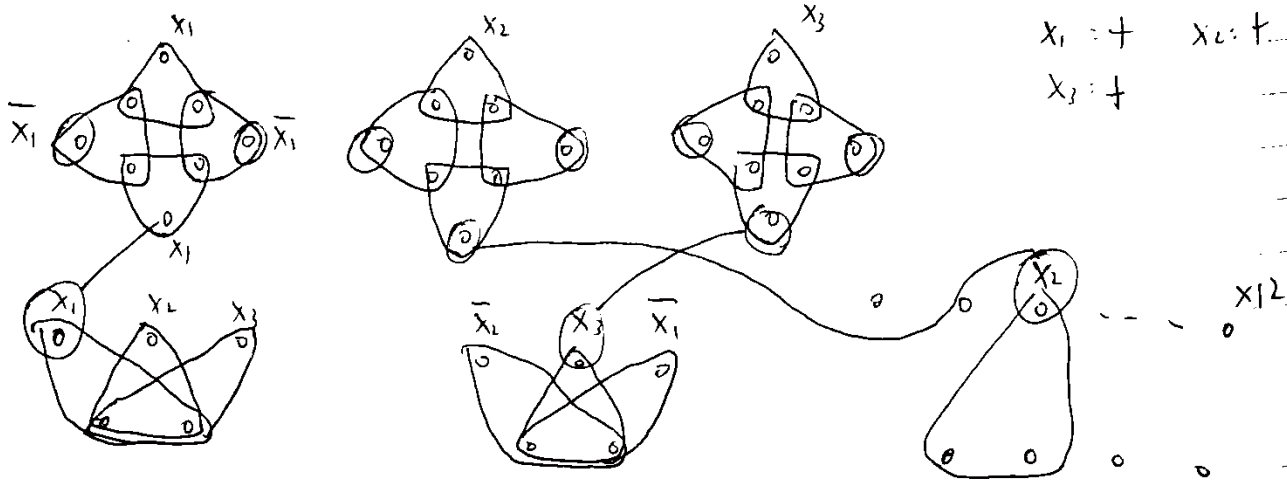
重复 $2mn$ 次. 共 $2mn$ V_i, $\bar{V}_i$.

internal elements. $g_1[k]$, $g_2[k]$, $1 \le k \le m(n-1)$

external elements. $V_i[j]$, $\bar{V}_i[j]$. $j \leftarrow [m]$, $i \leftarrow [n]$

a, b; S_1, S_2; g_1, g_2 : different sex
all V_i, $\bar{V}_i$: third sex

Example. $(X_1 \vee X_2 \vee X_3) \wedge (\bar{X}_2 \vee X_3 \vee \bar{X}_1)$. $X_3 = T$ suffices.



Lecture 8

Oct. 31 2024

Randomized Algorithms

Consider a random variable X with mean μ and some deviation $\lambda > 0$.
$Pr[X \geq \mu+\lambda]$: ~~upper~~ upper tail. $Pr[X \leq \mu-\lambda]$: lower tail.

Markov's inequality. Let X be a non-negative random variable and $\lambda > 0$, then
$Pr[X \geq \lambda] \leq \mathbb{E}[X]/\lambda$

Proof: $\mathbb{E}[X] = \underset{>0}{\mathbb{E}[X \mid X<\lambda]}\ \underset{>0}{Pr[X<\lambda]} + \underset{\geq\lambda}{\mathbb{E}[X \mid X \geq \lambda]}\, Pr[X \geq \lambda]$

$\geq \lambda\, Pr[X \geq \lambda]$.

Chebyshev's inequality. For any random variable X with mean μ and variance σ^2,
$Pr[|X-\mu| \geq \lambda] \leq \sigma^2/\lambda^2$

~~Proof: Pr[|X-μ|~~

Proof: $Pr[|X-\mu| \geq \lambda] = Pr[\underbrace{(X-\mu)^2}_{\to Y.} \geq \lambda^2]$, $\mathbb{E}[Y] = \sigma^2$

$\Rightarrow Pr[Y \geq \lambda^2] \leq \mathbb{E}[Y]/\lambda^2$ (Markov).

$= \sigma^2/\lambda^2$.

Example 1. Coin flip. Let $X_1, X_2, \ldots, X_n$ be iid (indep. ident. distr) Bernoulli random variables with $Pr[X_k = 0] = 1-p$ and $Pr[X_k = 1] = p$. Let $S_n = \sum X_i \sim Bin(n,p)$.
$\mathbb{E}[S_n] = np$, $Var[S_n] = np(1-p)$

Apply Markov's. $Pr[S_n - pn \geq \varepsilon n] \leq \frac{pn}{pn+\varepsilon n} = \frac{1}{1+\varepsilon/p}$. For $p = \frac{1}{2}$, $\frac{1}{1+2\varepsilon} \approx 1 - O(\varepsilon)$ for small $\varepsilon > 0$.

Apply Chebyshev's. $Pr[|S_n - pn| \geq \varepsilon n] \leq \frac{np(1-p)}{\varepsilon^2 n^2} < \frac{p}{\varepsilon^2 n}$.

Example 2. Balls & Bin. Throw n balls uniformly and randomly independently into n bins. Let L_i = # of balls in bin i. $\mathbb{E}[L_i] = 1$. $Var[L_i] = n \cdot \frac{1}{n}(1-\frac{1}{n}) = 1 - \frac{1}{n}$

Markov's. $Pr[L_i \geq 1+\lambda] \leq \frac{1}{1+\lambda} \approx \frac{1}{\lambda}$

for events $A_1, A_2, \ldots, A_n$

$\Pr[\bigcup_{i=1}^n A_i] \le \sum_{i=1}^n \Pr[A_i]$

Chebyshev's. $\Pr[|L_i - 1| \ge \lambda] \le \frac{1 - 1/n}{\lambda^2} \approx \frac{1}{\lambda^2}$.

Set $\lambda = 2\sqrt{n} \Rightarrow \Pr[|L_i - 1| \ge 2\sqrt{n}] \le \frac{1-1/n}{4n}$. n bins union bound,

$\Pr[\exists i, |L_i - 1| \ge 2\sqrt{n}] \le n \cdot \frac{1}{4n} = \frac{1}{4}$. $\Pr[\forall i, |L_i - 1| \le 2\sqrt{n}] \ge 1 - \frac{1}{4} = \frac{3}{4}$.

Example 3. Random walk. Start at the origin, and at each step, move a unit dist. either left or right uniformly, randomly and independently. What is the position after n steps.

Rademacher random variable. $X_i = \begin{cases} +1, & \text{w.p.} = 1/2, \\ -1, & \text{w.p.} = 1/2. \end{cases}$

$S_n = \sum X_i$. $\mu = 0$. $\sigma^2 = n$. Chebyshev. $\lambda = t\sigma = t\sqrt{n}$. $\Pr[S_n > t\sqrt{n}] \le \frac{1}{t^2}$.

Theorem 1. ($2k$-th order Moment inequality). Let $k \in \mathbb{Z}_{\ge 0}$. For any random variable X having mean μ, and finite moments up to $2k$ (an even number),

$$\Pr[|X - \mu| \ge \lambda] \le \frac{\mathbb{E}[(X-\mu)^{2k}]}{\lambda^{2k}}$$

Proof. Let $Y = (X-\mu)^{2k}$. $\Pr[|X-\mu|^{2k} \ge \lambda^{2k}] = \Pr[Y \ge \lambda^{2k}] \le \frac{\mathbb{E}[Y]}{\lambda^{2k}}$

Example. Random walk. $S_n = \sum X_i$.

$\mathbb{E}[S_n^4] = \mathbb{E}[(\sum X_i)^4] = \mathbb{E}[\sum X_i^4 + 4\sum X_i^3 X_j + 6\sum X_i^2 X_j^2 + 12\sum X_i^2 X_j X_k + 24 \sum X_i X_j X_k X_\ell]$

$= \sum \mathbb{E}[X_i^4] + 4\sum \mathbb{E}[X_i^3 X_j] + 6\mathbb{E}[X_i^2 X_j^2] + 12\sum \mathbb{E}[X_i^2 X_j X_k] + 24 \sum \mathbb{E}[X_i X_j X_k X_\ell]$

$n \times 1$; $\mathbb{E}[X_i^3]\mathbb{E}[X_j] = 0$; $6\binom{n}{2}$; $= 0$; $= 0$

$= n + 6\binom{n}{2}$.

$\Pr[|S_n| \ge t\sqrt{n}] \le \frac{\mathbb{E}[S_n^4]}{t^4 n^2} = \frac{n + 6\binom{n}{2}}{t^4 n^2} = \frac{\Theta(1)}{n^2}$.

Theorem (Hoeffding's inequality). Let $X_1, X_2, \ldots, X_n$ be n independent random variables taking values in $[0, 1]$. Let $S_n = \sum X_i$ with mean $\mu = \mathbb{E}[S_n] = \sum \mathbb{E}[X_i]$. Then for any $\lambda \ge 0$, upper tail $\Pr[S_n \ge \mu + \lambda] \le \exp\left(-\frac{\lambda^2}{2\mu + \lambda}\right)$

Laplace transform. $f_n : x \mapsto e^{tx}$, $t > 0$.

S_n

$$Pr[S_n \geq \mu+\lambda] = Pr[e^{tS_n} \geq e^{t(\mu+\lambda)}] \leq \frac{\mathbb{E}[e^{tS_n}]}{e^{t(\mu+\lambda)}} \quad (\text{markov})$$

$$= \frac{\prod_i \mathbb{E}[e^{tX_i}]}{e^{t(\mu+\lambda)}} \quad \text{Assuming } X_i \in \{0,1\}. \quad \mathbb{E}[X_i] = \mu_i = Pr[X_i = 1].$$

$$\mathbb{E}[e^{tX_i}] = 1 + \mu_i(e^t - 1) \Leftarrow \frac{Pr[X_i=0]}{(1-\mu)} e^{t\cdot 0} + \underset{\mu_i}{Pr[X_i=1]} \underset{e^t}{e^{t\cdot 1}}.$$

$$\leq \exp(\mu_i(e^t-1)) \Leftarrow 1+x \leq e^x$$

$$\leq \frac{\prod_i \exp(\mu_i(e^t-1))}{e^{t(\mu+\lambda)}} \leq \frac{\exp(\mu(e^t-1))}{e^{t(\mu+\lambda)}} = \exp(\mu(e^t-1) - t(\mu+\lambda)) \quad \text{求导, 令} = 0,$$

$$\Rightarrow t = \ln(1+\tfrac{\lambda}{\mu})$$

$$\leq \frac{e^\lambda}{(1+\lambda/\mu)^{\mu+\lambda}} \leq \exp\left(-\frac{\lambda^2}{2\mu+\lambda}\right)$$

Define $Y_i \sim Bernoulli(\mu_i)$. $Y_i = \begin{cases} 1, & w.p.\ \mu_i \\ 0, & w.p.\ 1-\mu_i \end{cases}$ $\mathbb{E}[Y_i]$ 也 $\mathbb{E}[X_i]$

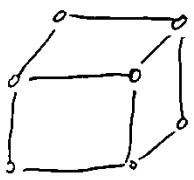
$f(x) = e^{tx}$ convex. $\forall t > 0$. $\ell(x) = (1-x) f(0) + x f(1)$. $\ell(x) \geq f(x)$, $\forall x \in [0,1]$.

$$\mathbb{E}[f(X_i)] \leq \mathbb{E}[\ell(X_i)] = \ell(\mathbb{E}[X_i]) = \ell(\mathbb{E}[Y_i]) = \mathbb{E}[\ell(Y_i)].$$

$$\mathbb{E}[e^{tX_i}] \leq \mathbb{E}[e^{tY_i}]$$

~~Example~~ Coin flipping.

Setting A: d-dimensional hypercube. Q_d with $n = 2^d$ nodes. each ~~table~~ labeled with a d-bit vector. Each node i has a single packet i, destined for node $\pi(i) \in [n]$ ↪ permutation.



- Packets move in synchronous rounds. Each edge is bi-directional
- Each edge e has a waiting queue W_e.

Goal. $O(d)$ rounds. delivers all $n = 2^d$ packets.

$\underset{src}{0\,0\,1\,0\,1\,0} \to 1\,0\,0\,1\,0\,1\,0 \to 1\,1\,0\,1\,0\,1\,0 \to \cdots \to \underset{dst}{1\,1\,0\,0\,0\,1\,1}$.

这样会冲突.

任何确定性算法不能在 $< \Omega(\sqrt{\frac{2^d}{d}})$ 步解决.

随机算法：① Each node i pick a random midpoint $R_i \in [n]$ and sends its packet to R_i
② After $5d$ rounds, all packets proceed to their final dst $\pi(i)$.

①：证明所有包在 $5d$ 轮大概率到中间节点。

Theorem. The random midpoint algorithm succeeds in $\leq 10d$ rounds. w.p. $\geq 1 - \frac{2}{n}$.
P_i: path from node i to R_i with bit flipping (page 37 bottom).
$S_i = \{j \neq i \mid$ path P_j shares some edge with $P_i\}$.

Claim 1. Any path P_i & P_j intersect in one contiguous segment.
Claim 2. Packet i reaches R_i by time $\leq |P_i| + |S_i|$

Proof. $P_i = \langle e_1, e_2, \ldots, e_\ell \rangle$.
① $|S_i| = 0$, ✓
②. Define lag: for any edge $e_k \in P_i$, we say every packet we at the beginning of time step t has $lag = t - k$.

$t=1$, e_1. $lag = 1 - 1 = 0$. t, e_k, $lag = t-k$. 若该步未被阻塞，下一步，
$t+1$, e_{k+1}. $lag = (t+1) - (k+1) = t-k$.

发送一个 $(t-k)$ token. invariant: 在每个时间步开始时候，一个标号为 L 的 token 如果还在 P_i 上的话，一定由某个 $lag = L$ 的 S_i 中的包携带。
若 $P_j \in S_i$，阻塞，一定有扣同的 lag

Lecture 9

Nov 7 2024

Hashing

1. Basics

Dictionary. keys $\in U$, $S \subset U$ $|S| = n \ll |U|$. Hash table A of size M.
supports operations: ins(x), query(q), del(x).
Desired properties:
- little collision. if $x \neq y$, $Pr[h(x) = h(y)]$ = small.
- Small range. $M \cong O(N)$
- small # of bits to store the hash function h.
- his eash to compute, ideally $O(1)$ computational overhead.

An ideal perfectly random hash function.
- For each $e \in U$, we consider a uniformly random location in $[M]$ and set $h(e)$ to that location.

(1) low collision. $a \neq b$. $Pr[h(a) = h(b)] = \frac{1}{M}$
(2) 每次映射完全独立. $Pr[h(e) = a \mid \bigcap_{a \in A} h(a) = \alpha_a] = \frac{1}{M}$.
(3). 然而，要存储 h. 就要存所有映射值查表 $|U| \log_2 M$ random 0/1 bits.
计算要查表. 也不一定快.

2. Universal Hashing

Definition A set of hash functions H where each $h \in H$ maps $U \to \{0, 1, \ldots, M-1\}$ is called universal if for all $x \neq y \in U$, we have

$$\Pr_{h \in H}[h(x) = h(y)] \leq \frac{1}{M}.$$

$$\Leftrightarrow \frac{|\{h \in H \mid h(x) = h(y)\}|}{|H|} \leq \frac{1}{M}$$

调用时，h 随机选定，此后不变.

Example.

H \ U	a	b
h_1	0	0
h_2	0	1

$Pr[\ldots] = \frac{1}{2}$. universal

	a	b	c
h_1	0	0	1
h_2	1	0	1

也是 universal hashing. 然而，对于 adversary，可以先给一个 a 看出用的 h_1 还是 h_2. 然后可以人构造碰撞.

universal 不能避免映射值的相关性. 每次之间不独立.

3. k-wise Universal / Independent Hashing

Definition. A family H of hash functions mapping U to $[M]$ is **k-wise universal / independent** if for any k distinct keys $x_1, x_2, \ldots, x_k \in U$ and any k locations $\alpha_1, \alpha_2, \ldots, \alpha_k \in [M]$ (not necessarily distinct),

$$\Pr_{h \leftarrow H}[h(x_1) = \alpha_1 \wedge h(x_2) = \alpha_2 \wedge \cdots \wedge h(x_k) = \alpha_k] \leq \frac{1}{M^k}$$

即：知道 k 个元素也不能更容易地猜出用的哪个 h.
知道 $k-1$ 个，对第 k 个也不知道.

Galois field. $GF(p^m)$. $\oplus, \ominus, \otimes, \oslash$.

$GF(2^m)$.
- elements. polynomials with coefficients in $GF(2)$. $\{0,1\}$, of degree $< m$.
- $\oplus$. addition of polynomials with coefficients added mod 2. (mod p).
- $\otimes$. multiply polynomials then mod a fixed irreducible polynomial of degree m.

Example. $GF(2^2)$
- elements. $\{0, 1, x, x+1\}$
- $\oplus$. $(x+1) \oplus (x+1) = 0$.
- $\otimes$ $f(x) = x^2+x+1$. $(x+1) \otimes (x+1) = x^2+2x+1 \bmod f(x) = x$.

k-wise independent hash function from $GF(p^m)$,
- Sampling h:
  (1) Sample random coefficients $c_0, c_1, \ldots c_{k-1} \in GF(p^m)$
  (2) hash function is then
  $$h(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{k-1} x^{k-1}$$

Claim. $\Pr[h(i_1) = a_1 \wedge h(i_2) \neq a_2 \wedge \cdots \wedge h(i_k) = a_k] \leq \frac{1}{|GF(p^m)|^k} = \frac{1}{p^{mk}}$.

Proof. $\Leftrightarrow \exists$ a unique solution $c_0, c_1, \ldots, c_{k-1} \in GF(p^m)$ s.t.

$$\begin{aligned}
c_0 + i_1 c_1 + i_1^2 c_2 + \cdots + i_1^{k-1} c_{k-1} &= a_1 \\
c_0 + i_2 c_1 + i_2^2 c_2 + \cdots + i_2^{k-1} c_{k-1} &= a_2 \\
&\vdots \\
c_0 + i_k c_1 + i_k^2 c_2 + \cdots + i_k^{k-1} c_{k-1} &= a_k
\end{aligned}$$

k equations, k unknowns.

$\Leftrightarrow$ columns of

$$\begin{pmatrix} 1 & i_1 & \cdots & i_1^{k-1} \\ 1 & i_2 & \cdots & i_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & i_k & \cdots & i_k^{k-1} \end{pmatrix}$$ are independent.

$$b_0 \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} + b_1 \begin{pmatrix} i_1 \\ \vdots \\ i_k \end{pmatrix} + \cdots b_{k-1} \begin{pmatrix} i_1^{k-1} \\ \vdots \\ i_k^{k-1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

然而，至多 k 个根

$\Rightarrow$ $i_1, i_2, \ldots, i_k$ are roots of the polynomial $b_0 + b_1 x + b_2 x^2 + \cdots + b_{k-1} x^{k-1} = 0$

$\Rightarrow$ $b_0 = b_1 = \cdots = b_{k-1} = 0$.

4. Searching for Nearest Neighbors (NN).

Locality Sensitive Hashing (LSH)

Problem setting. Given a set of n input points $P = \{p_1, p_2, \ldots, p_n\}$ in a metric space (X, d). We can preprocess it to create a data structure of "small" size in "small" time.

Now, queries $q_1, q_2, \ldots \in X$ arrives on time, for each query, we must return a point $\in P$ that is close to q.

ε-NN. Given a data set S of size n and a parameter $\varepsilon > 0$, return a s.t.

$$d(q, a) \leq (1+\varepsilon) \min_{x \in S} \{d(q, x)\}.$$

(ε, r)-NN. A fixed distance r to build a data structure NN_r to answer query q.

(1) if there exists some ~~point~~ point $x \in P$ with $d(q, x) \leq r$, the data structure must return a point a s.t. $d(q, a) \leq (1+\varepsilon) r$.

(2). if all points $x \in P$ has $d(q, x) > r$, the data structure can either return a point a with $d(q, a) \leq (1+\varepsilon) r$ or say NO.

Theorem. For any $\varepsilon \in (0, 1)$ and data set S of n points, a solution to (ε, r)-NN using $A(n, \varepsilon)$ preprocessing time & space and $Q(n, \varepsilon)$ query time can be used to give a solution to 3ε-NN with $O(A(n, \varepsilon) \frac{\log \Delta(S')}{\varepsilon})$ preprocessing time & space, and $O(Q(n, \varepsilon) \cdot \frac{\log \Delta(S')}{\varepsilon})$ query time. $\Delta(S) = d_{max}/d_{min}$. d_{max}/d_{min} are longest / shortest non-zero distance between points in S.

Proof. (1) for each r of the form, $r = \frac{d_{min}}{4}, \frac{d_{min}}{4}(1+\varepsilon), \frac{d_{min}}{4}(1+\varepsilon)^2, \cdots, \frac{d_{min}}{4}(1+\varepsilon)^i$ until $r > \frac{2d_{max}}{\varepsilon}$, we maintain a data structure for each (ε, r) - NN problem.

(2) When a query q arrives, we present it to each of these data structures and return the best of all answers

$$\log_{(1+\varepsilon)} \frac{2d_{max}/\varepsilon}{d_{min}/4} \approx O\left(\frac{\log \Delta(S')}{\varepsilon}\right)$$

② $\Delta(S')/\varepsilon$

② $\log_{(1+\varepsilon)} \frac{\Delta S}{\varepsilon} = \frac{\ln(\Delta(S')/\varepsilon)}{\ln(1+\varepsilon)} \approx \frac{\ln(\Delta(S')/\varepsilon)}{\varepsilon}$ $\in [\frac{\varepsilon}{2}, \varepsilon]$ for $\varepsilon \in (0,1)$.

3ε - NN.

(1) Suppose that the closest points in S' to the query is at distance D and $\frac{r}{1+\varepsilon} < D < r$
⇒ associated (ε, r) - NN will return a point at distance $\le (1+\varepsilon)r \le (1+\varepsilon)^2 D$
$\le (1+3\varepsilon)D$, for $\varepsilon < 1$.

(2) If D larger then the ~~largest~~ largest value r in our collection
⇒ any point in S' is an ε - NN answer.

(3) If $D < \frac{d_{min}/4}{1+\varepsilon}$ ⇒ The answer for $r = \frac{d_{min}}{4}$.

Locality Sensitive Hashing (LHS).

$d(x,y)$ close ⇒ $Pr[h(x) = h(y)] \ge$ large.

$d(x,y)$ far ⇒ $Pr[h(x) = h(y)] \le$ small.

(ε, r) - NN.

$d(x,y) \le r \Rightarrow Pr[h(x) = h(y)] \ge P_{close}$ (较大值) ≈ 1

$d(x,y) \ge (1+\varepsilon)r \Rightarrow$ --- $\le P_{far}$ (较小值) ≈ 0.

套路. (1) get a weak result. $P_{far} = P_{close} - \varepsilon$.

(2) Amplify the gap by parallel repetition. t independent hash functions $h'(x) = (h_1(x), h_2(x), \ldots h_t(x))$
$P_{far} \to (P_{far})^t = \frac{1}{n}$. 都冲突才算冲突.

(3) Serial repetition L hash tables. if x, y are close, they will collide in expected $\ge$ $L \cdot (P_{close})^t \approx 1$. ~~total~~

Total $L \cdot t$ independent hash functions.

Example. Hamming distance k-bits binary string.

- want to distinguish distance $\le r$ from $\ge 2r$. $\varepsilon = 1$.
- $h(x) = x_i$ [value of pos i] $d(x,y) \le r$, $Pr[h(x) = h(y)] \ge 1 - r/k = P_{close}$
$d(x,y) \ge 2r$, $\le 1 - 2r/k = P_{far}$.

Parallel repetition t. $(1-2r/k)^t \le \frac{1}{n}$. $e^{-(2r/k)t} \le \frac{1}{n} \Rightarrow t \ge \frac{k \log n}{2r}$. $P_{close} = \frac{1}{\sqrt{n}}$.

Serial repetition. $L = \sqrt{n} \log n$ independent copies of data structures.

$\Rightarrow \Pr[h(x) = h(y) \text{ for some of the } L \text{ copies}] \ge 1 - (1 - \frac{1}{\sqrt{n}})^L \approx 1 - e^{-L/\sqrt{n}} = 1 - \frac{1}{n}$.

LSH for sets via the Jaccard distance.

Nov. 14 2024 Lecture 10

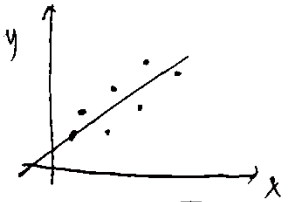
Sketching Algorithms.

1. Approximate Linear Squares Regression.

Linear least square regression. Given n data points $a_1, a_2, \dots, a_n \in \mathbb{R}^d$ and values $b_1, b_2, \dots, b_n \in \mathbb{R}$

Goal. Find a linear relationship (function) between a_i's and b_i's.

$$f(a_i) = b_i, \quad \forall i = 1 \dots n$$



Usually we cannot fit the curve perfectly because of noise in data. We want the curve to be as close to all the data points as possible, i.e.,

$$\min_x \sum_{i=1}^n (a_i^T x - b_i)^2 \iff \min_x \|Ax - b\|_2^2 \quad (1)$$

↳ if $x^* = \operatorname{argmin}_x \|Ax - b\|_2^2 \Rightarrow A^T A x^* = A^T b$ (2). normal equation

if columns of A are linear independent, $x^* = (A^T A)^{-1} A^T b$.

Complexity. $A \in \mathbb{R}^{n \times d}$, $A^T \in \mathbb{R}^{d \times n}$. $A^T A$: $O(nd^2)$. $(A^T A)^{-1}$: $O(d^3)$.
$(A^T A)^{-1} A^T$: $O(nd^2)$ $(A^T A) A^T b$ $O(nd)$
Total: assume $n \gg d$. $O(nd^2)$

Definition 1 (Approximate linear regression). Given $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, $\varepsilon > 0$, the ε- approximation regression problem is to find $x' \in \mathbb{R}^d$ s.t.

$$\|Ax' - b\|_2^2 \le (1+\varepsilon) \min_{x \in \mathbb{R}^d} \|Ax - b\|_2^2$$

Sketch - and - Solve Paradigm.

(1) Choosing a sketching matrix $S \in \mathbb{R}^{k \times n}$, $k \ll n$
(2) Compute $A' = SA$, $b' = Sb$
(3) Optimally solve (via normal equation)

$$x' = \operatorname{argmin}_x \|A'x - b'\|_2^2 = (A'^T A')^{-1} A'^T b'$$

$O(kd^2 + d^3)$. choose $k = O(\frac{d}{\varepsilon^2})$
$\Rightarrow O(\frac{d^4}{\varepsilon^2})$

Count-sketch matrix. Fix k, n and let $S \in \mathbb{R}^{k \times n}$ be defined as follows.

(1) Pick a 2-wise independent hash function $h: [n] \to [k]$ and
(2) pick a 4-wise independent hash function $s: [n] \to \{-1, 1\}$.

Define $S_{i,j} = \begin{cases} s(j), & \text{if } h(j) = i, \\ 0, & \text{otherwise} \end{cases}$

S 每列仅一个非零元. 由 h 决定在哪个行. 由 s 决定它是 -1 或 1.

Claim $A' = SA$ can be computed in $O(nnz(A))$. $nnz(A)$ = # of non-zero elements in A.

Proof (1) for any column vector $v \in \mathbb{R}^n$, Sv in $O(nnz(v))$.

(2) $A' = SA$. A has d such column vectors $\in \mathbb{R}^{n \times d}$ $\quad y_{h(i)} \leftarrow y_{h(i)} + s(i) v_i$ □

Definition. Fix any matrix A, let $V \subset \mathbb{R}^n$ be the column space of A. Then a matrix $S \in \mathbb{R}^{k \times n}$ is an ε- ~~subspace~~ embedding for V if $\forall x \in V$,

$$\|Sx\|_2 \in (1 \pm \varepsilon)\|x\|_2$$

or equivalently, $\forall x \in \mathbb{R}^d$,

$$\|SUx\|_2 \in (1 \pm \varepsilon)\|Ux\|_2$$

where $U \in \mathbb{R}^{n \times d}$ is an orthonormal basis for the column space of A.

Claim. If S is a subspace embedding for $[A; b]$, then the sketch-and-solve paradigm solves the ε- approximate regression problem.

Proof. $\forall x \in \mathbb{R}^d$, $y = Ax - b \in$ col-span $[A; b] \Rightarrow \|Sy\|_2 \in (1 \pm \varepsilon)\|y\|_2$

$\Leftrightarrow \|S(Ax - b)\|_2 \in (1 \pm \varepsilon)\|Ax - b\|_2$

$x^* = \arg\min_x \|Ax - b\|_2$. $\quad \|S(Ax - b)\|_2 \le \|S(Ax^* - b)\|_2 \in (1 + \varepsilon)\|Ax^* - b\|_2$.

对 A', b' 的最优解. 当然比 x^* 对 A', b' 更好.

□

Lemma (Approximate matrix product) Let $S \in \mathbb{R}^{k \times n}$ be a sketching matrix s.t. $k \ge \Omega(\frac{1}{\varepsilon^2 \delta})$. Let $A^{n \times d}$, $B^{n \times d}$ be any two matrices. Let $A^* = SA$, $B^* = SB$

$\Rightarrow \Pr[\|A^{*T}B^* - A^TB\|_F \le \varepsilon\|A\|_F\|B\|_F] \ge 1 - \delta$

$\|A\|_F = (\sum_{i,j} A_{ij}^2)^{\frac{1}{2}}$. notation $\delta_{i,j} = \begin{cases} 1, & \text{if } S_{i,j} \neq 0, \\ 0, & \text{otherwise,} \end{cases}$ $\quad \sigma(j) \in \{\pm 1\}$.

Proof. Let $C^* = A^{*T}B^*$. ① $\mathbb{E}[\|C^* - A^TB\|_F^2] \le \frac{2\sum_{u,u'}\|A_u\|_2^2\|B_{u'}\|_2^2}{k}$

② Apply Markov's inequality to upperbound the probability that $\|C^* - A^TB\|_F^2$ is too much larger than its expectation.

if $k \geq \frac{2}{\varepsilon^2 \delta}$, $\Pr[\|C^* - A^T B\|_F^2 \geq \varepsilon^2 \|A\|_F^2 \|B\|_F^2] \leq \delta$

$C^*_{u,u'} = \sum_{t=1}^{k} \sum_{i,j \in [n]} \sigma(i)\sigma(j)\,\delta_{t,i}\,\delta_{t,j}\,A_{i,u}\,B_{j,u'}$

$SB_{t,u'} = \sum_{j \in [n]} S_{t,j}\,B_{j,u'} = \sum_{j \in [n]} \delta_{t,j}\,\sigma(j)\,B_{j,u'}$

$(SA_{t,u})^T = \sum_{j \in [n]} S_{t,i}\,A_{i,u} = \sum_{i \in [n]} \delta_{t,i}\,\sigma(i) A_{i,u}$.

$= \underbrace{\sum_{t=1}^{k} \sum_{i \neq j \in [n]} \sigma(i)\sigma(j)\,\delta_{t,i}\,\delta_{t,j}\,A_{i,u}\,B_{j,u'}}_{\text{expectation} = 0} + (A^T B)_{u,u'}$ ↘ $i=j$.

$\sigma(i), \sigma(j)$ are independent for $i \neq j$. $\mathbb{E}[\sigma(i)\sigma(j)] = \underbrace{\mathbb{E}[\sigma(i)]}_{=0}\underbrace{\mathbb{E}[\sigma(j)]}_{=0} \Rightarrow \mathbb{E}[C^* - A^T B] = 0$

$(C^* - A^T B)^2_{u,u'} = \sum_{t_1,t_2=1}^{k} \sum_{i_1 \neq j_1 \in [n],\, i_2 \neq j_2 \in [n]} \sigma(i_1)\sigma(i_2)\sigma(j_1)\sigma(j_2)\,\delta_{t_1,i_1}\delta_{t_1,j_1}\delta_{t_2 i_2}\delta_{t_2 j_2}$
$A_{i_1,u} A_{i_2,u} B_{j_1,u'} B_{j_2,u'}$

$\mathbb{E}[\sigma^2(i_1)\sigma(i_2)\sigma^2(j_1)\sigma(j_2)]$

(1) $j_1 = j_2$, $j_1 = j_2$ ⟶ (2) $i_1 = j_2$, $i_2 = j_1$ ⟶ …

- ① $t_1 \neq t_2$. 0
- ② $t_1 = t_2$. $\mathbb{E}[\delta^2_{t_1,i_1}\,\delta^2_{t_1,j_1}]$ ← 2-wise indep.
$= \mathbb{E}[\delta^2_{t_1,i_1}]\,\mathbb{E}[\delta^2_{t_1,j_1}] = \underbrace{\mathbb{E}[\delta_{t_1,i_1}]}_{\frac{1}{k}}\underbrace{\mathbb{E}[\delta_{t_1,j_1}]}_{\frac{1}{k}}$

$= \frac{1}{k^2}$

$\mathbb{E}[(C^* - A^T B)^2_{u,u'}] = \mathbb{E}[\sum_{t_1=1}^{k} \delta^2_{t_1,i_1}\,\delta^2_{t_1,j_1}\,A^2_{i_1,u}\,B^2_{j_1,u'}] \leq \frac{A^2_{i_1,u}\,B^2_{j_1,u'}}{k}$

Summing over all possible i_1, j_1,

$\leq \frac{\|A_u\|_2^2 \|B_{u'}\|_2^2}{k}$

Cauchy - Schwartz inequality Let $u, v \in \mathbb{R}^n$. $|\langle u, v\rangle| \leq \|u\|_2 \|v\|_2$. If A is a matrix, $\|Av\|_2 \leq \|A\|_F \|v\|_2$.

Theorem (Subspace embedding). Let V be any fixed d-dimensional subspace. If $k \geq \Omega(\frac{d^2}{\varepsilon^2 \delta})$, then with probability $\geq 1-\delta$, $\forall x \in V$ we have $\|Sx\|_2 \in (1 \pm \varepsilon)\|x\|_2$

Proof. Let $U \in \mathbb{R}^{n \times d}$ be an orthonormal basis for V. ① $U^T U = I_d$, ② $\|U\|_F^2 = d$.

③ $\forall x \in \mathbb{R}^d$, $\|Ux\|_2^2 = \|x\|_2^2$.

Let $U^* = SU$ be the sketch, let $\varepsilon' = \varepsilon/d$. $k \geq \Omega\left(\frac{d^2}{\varepsilon'^2 d}\right) \Leftrightarrow k \geq \Omega\left(\frac{1}{\varepsilon^2\delta}\right)\varepsilon$

$\Rightarrow Pr[\|U^{*T}U^* - U^TU\|_F \leq \varepsilon'\|U\|_F^2] \geq 1-\delta$

$\forall x \in \mathbb{R}^d$, $\|U^*x\|_2^2 - \|Ux\|_2^2 = (U^*x)^T(U^*x) - (Ux)^T(Ux)$

$= x^T(U^{*T}U^* - I_d)x \leq \|x^T\|_2 \|U^{*T}U^* - I_d\|_F \|x\|_F$

$\leq \varepsilon\|x\|_2^2$

$\Rightarrow \|U^*x\|_2^2 \leq (1+\varepsilon)\|x\|_2^2$.

□

Nov 21 2024 Lecture 11

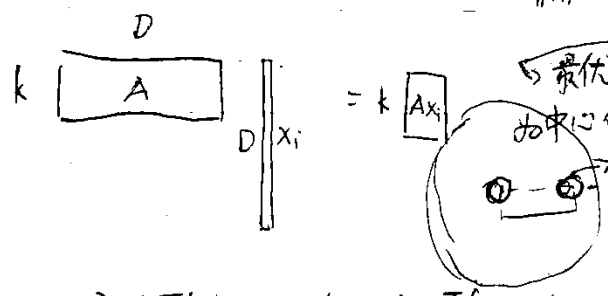
1. Dimension Reduction and JL Lemma

Motivation. from data science. Suppose. $x_1, \dots, x_n \in \mathbb{R}^D$ data points in high dimensional space, how can we compress these data points (map into lower dimensional space), while keeping pairwise relations, e.g., Euclidean distance $\|x_i - x_j\|_2$

若要精确保持，则最多降到 $n-1$ 维.

Theorem 1. (Johnson - Lindeastrauss) Given any set of n points $X = \{x_1, x_2, \dots, x_n\}$, $x_i \in \mathbb{R}^D$, there exists a map (projection matrix) $A \in \mathbb{R}^{k \times D}$. $A: \mathbb{R}^D \to \mathbb{R}^k$, $k = O(\frac{\log n}{\varepsilon^2})$ s.t.

$$1-\varepsilon \le \frac{\|Ax_i - Ax_j\|_2^2}{\|x_i - x_j\|_2^2} \le 1+\varepsilon, \quad i,j \in [n]$$



最优：假定 $\mathbb{R}^D$ 有两两距离 1 的点，到 k 维中每个点为中心作 $\frac{1}{2}(1-\varepsilon)$ 小球又作 $1+\varepsilon+\frac{1}{2}(1-\varepsilon)$ 大球，在大球内 $\Rightarrow k = O(\dots)$.

Lemma 2. (JL Lemma. 1982) If each entry of $A \sim \frac{N(0,1)}{\sqrt{k}}$ (每个元素独立取，正态分布), $k = O(\varepsilon^{-2} \log \delta^{-1})$, then for any given unit vector $x \in \mathbb{R}^D$,

$$Pr[\|Ax\|_2^2 \in (1 \pm \varepsilon)] \ge 1 - \delta$$

Proof of Theorem 1. Set $\delta = 1/n^3$, $k = \frac{\log n}{\varepsilon^2}$, for any $x_i, x_j \in X$, apply Lemma 2 on $\frac{x_i - x_j}{\|x_i - x_j\|_2}$, $Pr[\|A \frac{x_i - x_j}{\|x_i - x_j\|_2}\|_2^2 \in (1 \pm \varepsilon)] \ge 1 - \frac{1}{n^3}$.

$\Leftrightarrow Pr[\frac{\|Ax_i - Ax_j\|_2^2}{\|x_i - x_j\|_2^2} \in (1 \pm \varepsilon)] \ge 1 - \frac{1}{n^3}$ 不在 $(1 \pm \varepsilon)$ 的概率 $\le \frac{1}{n^3}$

共有 $\binom{n}{2}$ 个 (x_i, x_j) 点对，则由 union bound，所有点对中出现一个不在 $(1 \pm \varepsilon)$ 的概率 $\le \binom{n}{2} \frac{1}{n^3} < \frac{1}{n}$. □

Proof of Lemma 2. $x = e_1 = (1, 0, \dots, 0)^T$. $Ax = k$ [D] $\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} A_{11} \\ A_{21} \\ \vdots \\ A_{k1} \end{pmatrix} = \frac{1}{\sqrt{k}} \begin{pmatrix} N(0,1) \\ N(0,1) \\ \vdots \\ N(0,1) \end{pmatrix}$

$\sim \begin{pmatrix} N(0,\frac{1}{k}) \\ N(0,\frac{1}{k}) \\ \vdots \\ N(0,\frac{1}{k}) \end{pmatrix}$ $\|Ax\|_2^2 = N(0,\frac{1}{k})^2 + \cdots + N(0,\frac{1}{k})^2 = kN(0,\frac{1}{k})^2$.

$\mathbb{E}[\|Ax\|_2^2] = Var[\|Ax\|_2^2] + \mathbb{E}[\|Ax\|_2]^2$ ~~$= k\mathbb{E}$~~

$= k\mathbb{E}[N(0,\frac{1}{k})^2] = \underbrace{k\,Var[N(0,\frac{1}{k})]}_{=k\cdot\frac{1}{k}=1} + \underbrace{k\,\mathbb{E}[N(0,\frac{1}{k})]^2}_{0}$

$= 1$. 期望看起来对.

$x = (x_1, x_2, \ldots, x_D)$. $Ax = k$ [D] $\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix} = \frac{1}{\sqrt{k}} \begin{pmatrix} \square \\ \to N(0,1) \end{pmatrix}$

$\square = x_1 N(0,1) + x_2 N(0,1) + \cdots + x_D N(0,1)$

$= N(0, x_1^2) + \cdots + N(0, x_D^2)$

$= N(0, \underbrace{x_1^2 + x_2^2 + \cdots + x_D^2}_{=1}) = N(0,1)$

$\|Ax\|_2^2 \sim \frac{1}{k}(N(0,1)^2 + \cdots + N(0,1)^2) \in (1 \pm \varepsilon)$.

令 $y_i = N(0,1)$ indep. $Z = \frac{1}{k}\sum y_i$. → 变成增长快的东西.

$Pr[Z \geq 1+\varepsilon] = Pr[e^{tkZ} \geq e^{tk(1+\varepsilon)}]$ (Laplace)

$\leq \frac{\mathbb{E}[e^{tkZ}]}{e^{tk(1+\varepsilon)}}$ (Markov)

$= \frac{\mathbb{E}[\pi_i e^{ty_i^2}]}{e^{tk(1+\varepsilon)}} \overset{y_i \text{ indep}}{=} \frac{\pi_i \mathbb{E}[e^{ty_i^2}]}{e^{tk(1+\varepsilon)}}$

$\mathbb{E}[e^{ty_i^2}] = \int_{-\infty}^{+\infty} e^{tg^2} \frac{1}{\sqrt{2\pi}} e^{-\frac{g^2}{2}} dg = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{tg^2 - \frac{g^2}{2}} dg = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{-\frac{1-2t}{2}g^2} dg =$ ~~$\frac{1}{\sqrt{1-2t}}$~~

$= \frac{1}{\sqrt{1-2t}}$

$\Rightarrow \mathbb{E}[e^{ty_i^2}] = \frac{1}{\sqrt{1-2t}} \Rightarrow Pr[Z \geq e^{tk}] = \frac{\pi_i \mathbb{E}[e^{ty_i^2}]}{e^{tk(1+\varepsilon)}} = \left(\frac{1}{e^{t(1+\varepsilon)}\sqrt{1-2t}}\right)^k$. 要 $\leq e^{\varepsilon^2 k/8}$

$= e^{-\ln(2/\delta)}$.

$= \frac{\delta}{2}$

$e^{t(1+\varepsilon)} = 1 + t(1+\varepsilon) + O(t^2)$. $\sqrt{1-2t} = 1 - t - O(t^2)$.

$\textcircled{1}\times\textcircled{2} = 1 - t^2 + t\cdot\varepsilon + O(t^2)$ 要 $\geq e^{\varepsilon^2/8}$

令 $t = \frac{\varepsilon}{4}$.

Set $k = \frac{8}{\varepsilon^2}\ln\frac{2}{\delta}$

2. Estimating $\|f^t\|_2$ by Tug-of-War.

$U = \{e_1, e_2, \ldots, e_u\}$. add$(e_i)$, del$(e_j)$.
$f^t = (f_1^t, f_2^t, \ldots, f_u^t)$. $\|f^t\|_2 = \sqrt{\sum_{i=1}^u f_i^{t2}}$

① Pick a random hash function $h: U \to \{-1, +1\}$
② Maintain an int counter C (init $= 0$).
③ On each update, add i: $C \leftarrow C + h(i)$.
del i: $C \leftarrow C - h(i)$
④ On query (of $\|f^t\|_2$). return C^2.

Claim. $\mathbb{E}[C^2] = \|f^t\|_2^2$, $C = \sum_{i \in U} f_i h(i)$.

Proof. $\mathbb{E}[C^2] = \mathbb{E}[\sum_{i,j} f_i h(i) f_j h(j)] = \sum_{i,j} f_i f_j \mathbb{E}[h(i)h(j)]$
$= \sum_i f_i^2 \mathbb{E}[h(i)^2]$ $\frac{1}{2}(1)^2 + \frac{1}{2}(-1)^2 = 1$.
$= \sum_i f_i^2$
$= \mathbb{E}[h(i)]\mathbb{E}[h(j)] = 0$ if $i \neq j$. □

$\mathrm{Var}[C^2] = \mathbb{E}[(C^2)^2] - \mathbb{E}[C^2]^2$.
$\mathbb{E}[(C^2)^2] = \mathbb{E}[\sum_{i,j,k,l} h(i)h(j)h(k)h(l) f_i f_j f_k f_l]$
$= \sum_i f_i^4 \mathbb{E}[h(i)^4] + 6 \sum_{i \neq j} f_i^2 f_j^2 \mathbb{E}[h(i)^2 h(j)^2]$
$\mathrm{Var}[C^2] = \sum_i f_i^4 + 6 \sum_{i,j} f_i^2 f_j^2 - (\sum_i f_i^2)^2$
$= 4 \sum_{i \neq j} f_i^2 f_j^2 \leq 2\,\mathbb{E}[C^2]^2$.

$$\Pr[|C^2 - \mathbb{E}[C^2]| > \varepsilon \mathbb{E}[C^2]] \leq \frac{\mathrm{Var}[C^2]}{(\varepsilon \mathbb{E}[C^2])^2} \leq \frac{2}{\varepsilon^2}$$

$\mathbb{E}[\bar{C}] = \mathbb{E}[\frac{1}{k}\sum_i C_i] = \frac{1}{k}\sum_i \mathbb{E}[C_i] = \|f^t\|_2$
$\mathrm{Var}[\bar{C}] = \mathrm{Var}[\frac{1}{k}\sum_i C_i] = \frac{1}{k^2}\sum_i \mathrm{Var}[C_i] \leq \frac{2}{k^2}(\sum_i f_i)^2$.

$$\Rightarrow \Pr[|\bar{C} - \mathbb{E}[\bar{C}^2]| > \varepsilon \mathbb{E}[\bar{C}^2]] \leq \frac{\mathrm{Var}[\bar{C}^2]}{(\varepsilon \mathbb{E}[\bar{C}^2])^2} \leq \frac{2}{k\varepsilon^2} < 1$$

Theorem. (Tug-of-War sketching) Take a $k \times D$ matrix S, where columns are 4-wise independent $\{\frac{1}{\sqrt{k}}, -\frac{1}{\sqrt{k}}\}^k$-valued random variables. Then for $x, y \in \mathbb{R}^D$,
① $\mathbb{E}[\langle Sx, Sy\rangle] = \langle x, y\rangle$, ② $\mathrm{Var}[\langle Sx, Sy\rangle] = \frac{2}{k}\|x\|_2 \|y\|_2$.

$A, B \in \mathbb{R}^{n \times n}$. $A^* = AS^T = (SA^T)^T$. $B^* = SB$

$\Pr[\|C^* = A^*B^* - A^TB\|_F \le \varepsilon\|A\|_F\|B\|_F] \ge 1-\delta$?

$\mathbb{E}[(C^*-C)_{i,j}] = 0$ — let $Y_{i,j} = (C^*-C)_{i,j}$

$\mathbb{E}[Y_{i,j}]^2 = Var[Y_{i,j}] + \mathbb{E}[Y_{i,j}]^2$

$= \frac{2}{k}\|A_{i,*}\|_2^2 \|B_{*,j}\|_2^2$ ← ②

$\mathbb{E}[\|C^* - C\|_F^2] = \mathbb{E}[\Sigma_{i,j} Y_{i,j}^2] = \Sigma_{i,j} \mathbb{E}[Y_{i,j}^2] = \frac{2}{k}\Sigma_{i,j}\|A_{i,*}\|_F^2\|B_{*,j}\|_F^2$

let $k = \frac{2}{\varepsilon^2\delta} = \frac{2}{(2/\varepsilon^2\delta)}\|A\|_F^2\|B\|_F^2$

$\Pr[\sim]$

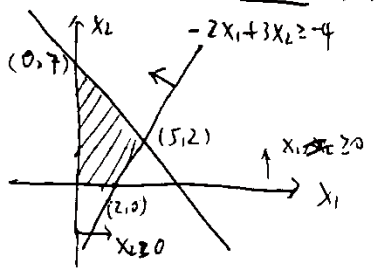
Dec. 5 2024 Lecture 13

Linear Programming

1. Introduction

subject to.

max $5x_1+x_2$, s.t. $x_1+x_2 \le 7$, $-2x_1+3x_2 \ge -4$, $x_1, x_2 \ge 0$.



$\max (5,1)^T (x_1, x_2)$

Example1. Max flow in $G=(V,E)$. from $s \to t$.

$\max \sum_{v:(s,v)\in E} f(s,v)$ s.t. $\sum_{u:(u,v)\in E} f(u,v) = \sum_{w:(v,w)\in E} f(v,w)$

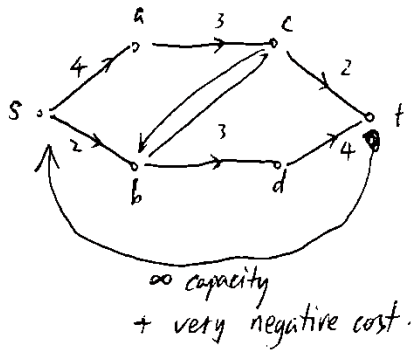
{conservative constraint}. $f(u,v) \le cap(u,v)$, $f(u,v) \ge 0$

{capacity constraint}

Example2. min-cost max flow

$\min \sum_{(u,v)\in E} \$(u,v) f(u,v)$ s.t. $\sum_{u\in V} f(s,u) = f_{max}$.

用一个 LP 解决 Example 1, 2.

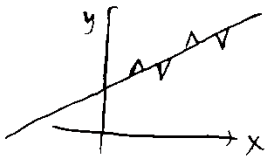


$\min \sum_{(u,v)\in E} \$(u,v) f(u,v)$. s.t.

$0 \le f(u,v) \le cap(u,v)$,

$\sum_{u\in V} f(u,v) = \sum f(v,w)$, $\forall v \in V$.

L_2 regression.



$\min_{a,b} \sum_i |(ax_i+b)-y_i|$ $\Rightarrow$ $\min z_i$ s.t $ax_i+b-y_i \le z_i$, $-z_i \le ax_i+b-y_i$, $z_i \ge 0$.

$\min \vec{c}^T\vec{x}$, s.t.	$\Rightarrow$	$\min \vec{c}^T\vec{x}$, s.t.	standard Form	max ~~~
$\vec{a}_1\vec{x} \geq \vec{b}_1$		$A\vec{x} \geq \vec{b}$.		$A\vec{x} \leq \vec{b}$.
$\vdots$		$\vec{x} \geq 0$		
$\vec{a}_m\vec{x} \geq \vec{b}_m$				
$\vec{x} \geq \vec{0}$				

Common trick to convert to standard

(1) Inequality the wrong way. $(\times(-1))$.

(2) Equality $x_1 - x_2 = q \iff \begin{cases} x_1 - x_2 \geq q \\ x_1 - x_2 \leq q \end{cases}$

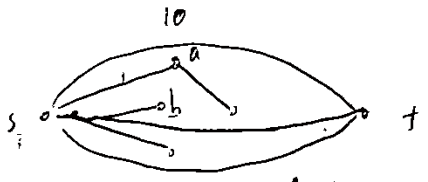
(3) x_i unbounded $\Rightarrow$ $x_i = x_i^+ - x_i^-$, $x_i^+, x_i^- \geq 0$.

Example 4. Shortest path from $s \to t$ (1 unit flow)

$\min \sum_{e\in E} len(e) f(e)$, s.t. $\sum_{e \text{ leaving } s} f(e) = 1$,

$\sum_{e \text{ into } v} f(e) = \sum_{e' \text{ out of } v} f(e')$, $\forall v \in V \setminus \{s,t\}$,

$f(e) \geq 0$, $\forall e \in E$.



*下拉绳子，从 s,t 两头到一根拉不动为止.

$\max d(t)$, s.t. $d(v) \leq d(u) + len(\overrightarrow{u,v})$, $\forall (u,v) \in E$. 三角不等式. $d(s) = 0$

2. Duality

$\max\ 2x_1 + 3x_2$
s.t.
$4x_1 + 8x_2 \leq 12$
$2x_1 + x_2 \leq 3$
$3x_1 + 2x_2 \leq 4$
$x_1, x_2 \geq 0$.

$2x_1 + 3x_2 \leq 4x_1 + 8x_2 \leq 12$

$2x_1 + 3x_2 \leq \frac{1}{2}(4x_1 + 8x_2) \leq 6$

$2x_1 + 3x_2 \leq \frac{1}{3}\left[\underbrace{(4x_1 + 8x_2)}_{1st \leq 12} + \underbrace{(2x_1 + x_2)}_{2nd \leq 3}\right]$

≤ 5

~~$y_1 + y_2$~~

$y_1\underbrace{(1st)}_{\leq 12} + y_2\underbrace{(2nd)}_{\leq 3} + y_3\underbrace{(3rd)}_{\leq 4}$

$\leq (12y_1 + 3y_2 + 4y_3)$

s.t.
$\begin{cases} 4y_1 + 2y_2 + 3y_3 \geq 2 \\ 8y_1 + y_2 + 2y_3 \geq 3 \end{cases}$
$\min$

$\max \vec{c}^T\vec{x}$	$\min \vec{y}^T\vec{b}$
s.t. $A\vec{x} \leq \vec{b}$	s.t. $\vec{y}^TA \geq \vec{c}^T$
$\vec{x} \geq \vec{0}$	$\vec{y} \geq 0$.
Primal LP	Dual LP.

Weak Duality.

Strong Duality. If $\vec{x}$ is an OPT. sol. to the primal LP & $\vec{y}$ is a OPT. sol. to the dual LP, then $\vec{c}^T\vec{x} = \vec{y}^T\vec{b}$.

Primal \ Dual	Infeasible	Feasible & Bounded	Unbounded
Inf	✓	✗	✓
F&B	✗	✓	✗
Unb	✓	✗	✗

3. Minimax optimality in zero-sum game.

m \ n	R	P	S
R	0	-1	+1
P	+1	0	-1
S	-1	+1	0

M

(1/3, 1/3, 1/3)

row player picks $\vec{p} \in \Delta_m$.
col player picks $\vec{q} \in \Delta_n$.

Expected payoff $= \sum_{i,j} p_i q_j M_{ij} = \vec{p}^T M \vec{q}$

$\min_{\vec{p}} \vec{p}^T M \vec{q}$, $c(\vec{p}) = \max_{j \in [n]} \vec{p}^T M \vec{e_j}$.

(0, 0, 0, 0, 0)

maximize $\max \hat{c}(\vec{p})$. $\vec{p} \in \Delta_m$. $R(\vec{q}) = \max_{i \in [m]} \vec{e_i}^T M \vec{q}$

4. Seidel's algorithm.

① $O(n)$ expected time. Solve a 2Dimensional LP with n constraints.

②. d-dimensional LP (d variables) with n constraints: $O(d!\,n)$

Weighted Majority Algorithm (WM)

w_i^t, $\forall i \in [N]$, $w_i^0 = 1$.

$a^t \leftarrow \text{argmax}_{u \in U} \sum w_i^t$ $\quad$ i: expert i predicts u.

o^t.

$$w_{i+1}^{t+1} = w_i^t \cdot \begin{cases} 1, & \text{if } i \text{ correct}, \\ 1-\varepsilon, & \text{if } i \text{ wrong}. \end{cases}$$

Theorem: For any seq. predictions, #mistakes made by the WM algorithm is $\le 2(1+\varepsilon) m^* + O(\frac{\log N}{\varepsilon})$, $\varepsilon \in (0, \frac{1}{2})$.

Proof. $\Phi^t = \sum_{i \in [N]} w_i^t$, $\Phi^0 = N$, $\Phi^{t+1} \le \Phi^t$.

WM algorithm. t_i predict wrong,

$$\Phi^{t+1} = \sum_{i \text{ wrong}} w_i^{t+1} + \sum_{i \text{ correct}} w_i^{t+1}$$
$$= \frac{1}{2} \sum_{i \text{ wrong}} w_i^t + \downarrow$$
$$= \Phi^t - \frac{1}{2} \sum_{i \text{ wrong}} w_i^t \le (1 - \frac{\varepsilon}{2}) \Phi^t.$$

After T steps, best expert makes m^* mistakes

$$(1-\varepsilon)^{m^*} = w_i^T \le \Phi^T \le \Phi^0 (1 - \frac{\varepsilon}{2})^M.$$

WM algorithm $\quad$ M

$$m^* (-1) \le \log_2 N + M \log_2 \frac{3}{4}$$
$$M \log_2 \frac{4}{3} \le m^* + \log_2 N$$
$$M \le \frac{m^* + \log_2 N}{\log_2 \frac{4}{3}}$$

Randomized Weighted Majority (RWM).

$$\Pr_{u \in U}[\text{action } u \text{ is predicted}] = \frac{\sum_{i: \text{expert } i \text{ predicts } u} w_i^t}{\sum_i w_i^t}$$

Theorem. For any seq predictions, # mistakes made by the ~~RWM~~ RWM algorithm $\mathbb{E}[M] \le (1+\varepsilon) m^* + O(\frac{\log N}{\varepsilon})$, $\varepsilon \in (0, \frac{1}{2})$

Proof. $\Phi^t = \sum_{i \in [N]} w_i^t$, $\Phi^0 = N$, $\Phi^{t+1} \le \Phi^t$, $\forall t$.

$$F^t = \frac{\sum_{i \text{ wrong}} w_i^t}{\sum_i w_i^t} \Rightarrow \mathbb{E}[M] = \sum_{t \in [T]} F^t.$$

$$\Phi^{t+1} = \Phi^t \left(\underbrace{(1-F^t)}_{\text{correct}} + \underbrace{F^t}_{\text{wrong}} (1-\varepsilon) \right) = \Phi^t (1 - \varepsilon F^t)$$

$$(1-\varepsilon)^{m^*} \le \Phi^T \le \Phi^0 \prod_{t=1}^{T} \underbrace{(1-\varepsilon F^t)}_{\le e^{-\varepsilon F^t}} \le N e^{-\varepsilon \sum F^t} = N e^{-\varepsilon \mathbb{E}[M]}.$$

1. The algorithm produces a vector of probabilities
   $\vec{p}^t = (p_1^t, p_2^t, \dots, p_N^t) \in \Delta_N$, s.t. $\sum_{i\in[N]} p_i^t = 1$.
2. Adversary generates a loss vector
   $\vec{l}^t = (l_1^t, l_2^t, \dots l_N^t) \in [-1,1]^N$
3. Each round, algorithm incurs loss $= \langle \vec{l}^t, \vec{p}^t \rangle$
4. $w_i^0 = 1$, $\forall i \in [N]$
5. t step , $p_i^t = \frac{w_i^t}{\sum_j w_j^t}$, $\sum_j w_j^t =: \Phi^t$
6. $w_i^{t+1} = w_i^t \exp(-\varepsilon l_i^t)$.

Hedge. 对冲再证. Φ^0 并不会↓.

Theorem: For any seq predictions, # mistakes made by the Hedge
$\sum_{t=1}^{T} \langle \vec{p}^t, \vec{l}^t \rangle \le \sum_{t=1}^{T} l_i^t + \varepsilon + \frac{\ln N}{\varepsilon}$, $\varepsilon \in (0, \frac{1}{2})$

Proof: $\Phi^t = \sum_{i\in[N]} w_i^t$. $\Phi^0 = N$

$\Phi^{t+1} = \sum_i w_i^{t+1} = \sum_i w_i^t \cdot e^{-\varepsilon l_i^t}$ $\{e^x \le 1 + x + x^2 \text{ for } x \in [-1,1]\}$

$\le \sum_i w_i^t (1 - \varepsilon l_i^t + \varepsilon^2 (l_i^t)^2)$ $\{l_i^t \le 1\}$

$\le \sum_i w_i^t (1+\varepsilon^2) - \varepsilon \sum_i w_i^t l_i^t$

$\le (1+\varepsilon^2) \frac{\sum_i w_i^t}{\Phi^t} - \varepsilon \Phi^t \frac{\sum_i p_i^t l_i^t}{\langle \vec{p}^t, \vec{l}^t \rangle}$

$\Rightarrow \Phi^{t+1} \le \Phi^t (1 + \varepsilon^2 - \varepsilon \langle \vec{p}^t, \vec{l}^t \rangle) \le \Phi^t e^{\varepsilon^2 - \varepsilon \langle \vec{p}^t, \vec{l}^t \rangle}$ $\{1 + x \le e^x\}$.

$e^{-\varepsilon \sum_{t=1}^{T} l_i^t} = w_i^{t+1} \le \Phi^{t+1} \le \dots$

$\le \Phi^1 \exp(\varepsilon^2 T - \varepsilon \sum_{t\in[T]} \langle \vec{p}^t, \vec{l}^t \rangle$

$\le \frac{\Phi^1}{N}$

ln

$\varepsilon T = \frac{\ln N}{\varepsilon} \Rightarrow \varepsilon = \sqrt{\frac{\ln N}{T}} \Rightarrow \frac{\text{regret}}{T} = 2\sqrt{\frac{(\ln N)\cdot T}{T}} = \frac{2\sqrt{\ln N}}{\sqrt{T}}$

Corollary 1: For $T \ge \frac{4 \ln N}{\varepsilon^2}$, Hedge algorithm's average loss
$\frac{1}{T} \sum_t \langle \vec{p}^t, \vec{l}^t \rangle \le \min_i \frac{1}{T} \sum_t l_i^t + \varepsilon$

Corollary 2: Let $\rho \ge 1$ & $\varepsilon \in (0, \frac{1}{2})$. For any seq of gain vectors $\vec{g}^1, \vec{g}^2, \dots, \vec{g}^T \in [-\rho, \rho]^N$
with $T \ge \frac{4\rho^2 \ln N}{\varepsilon^2}$, $\frac{1}{T} \sum_{t=1}^{T} \langle \vec{g}^t, \vec{p}^t \rangle \ge \max_{i\in[N]} \frac{1}{T} \sum_{t=1}^{T} \langle \vec{g}^t, \vec{e}_i \rangle - \varepsilon$.

不完全信息（现实中没有完全的损失信息预先知道）下的 Hedge algorithm.

$Pr[I^t = i] = q_i^t = \gamma \frac{1}{N} + (1-\gamma) p_i^t$.

$$\hat{l}_j^t = \begin{cases} l_j^t / q_j^t, & \text{if } j = a^t, \\ 0, & \text{if } j \neq a^t. \end{cases}$$

$\mathbb{E}[\hat{l}_i^t] = \frac{l_i^t}{q_i^t} \cdot q_i^t + 0 \cdot (1-q_i^t) = l_i^t$

$\hat{l}_j^t = l_j^t \cdot 1/q_j^t \in [-1,1] \leq \frac{N}{\gamma}$ ~~$[-1,1] \leq N/\gamma$~~ $[-\frac{N}{\gamma}, \frac{N}{\gamma}] [-\rho, \rho]$

$\Rightarrow \mathbb{E}[\sum_t \langle \vec{p}^t, \vec{l}^t \rangle] \leq \sum_t \hat{l}_i^t + \frac{N}{\gamma}(\varepsilon T + \frac{\ln N}{\varepsilon})$ {Corollary 2}.

$\Rightarrow \sum_t \langle \vec{p}^t, \vec{l}^t \rangle \leq \sum_t l_i^t + \frac{N}{\gamma}(\varepsilon T + \frac{\ln N}{\varepsilon})$

$\sum_t \langle \vec{p}^t, \vec{l}^t \rangle = (1-\gamma) \sum_t \langle \vec{p}^t, \vec{l}^t \rangle + \frac{\gamma}{N} \sum_t \langle \vec{1}, \vec{l}^t \rangle$

$\leq \sum_t l_i^t + \underbrace{\frac{N}{\gamma}(\varepsilon T + \frac{\ln N}{\varepsilon}) + \gamma T}_{\text{regret}}$

$\varepsilon = \sqrt{\frac{\ln N}{T}}$, $\gamma = \sqrt{N} (\frac{\ln N}{T})^{\frac{1}{4}}$

$\Rightarrow$ regret $= N^{\frac{1}{2}} (\ln N)^{\frac{1}{4}} T^{\frac{3}{4}} < T$, $\frac{\text{poly}(N)}{T^{1/4}} \to 0$.

Lecture 15

Dec 19 2024

Gradient Descent (GD)

Unconstrained setting $\min_{\vec{x}\in\mathbb{R}^n} f(\vec{x})$; constrained setting $\min_{\vec{x}\in K} f(\vec{x})$, $K\subset\mathbb{R}^n$. f 是凸出数, K 是凸集.

Definition (convex set) A set K is called convex if $\forall \vec{x},\vec{y}\in K$, $\lambda\vec{x}+(1-\lambda)\vec{y}\in K$, $\forall\lambda\in[0,1]$

Definition (convex function) A function $f: K\to\mathbb{R}$ ~~is called conf convex~~ defined on a convex set K is called convex if $\forall \vec{x},\vec{y}\in K$, $f(\lambda\vec{x}+(1-\lambda)\vec{y}) \le \lambda f(\vec{x}) + (1-\lambda) f(\vec{y})$, $\forall\lambda\in[0,1]$.

First-order condition for checking convexity of a function: A function $f: K\to\mathbb{R}$ is convex iff. $f(\vec{y}) \ge f(\vec{x}) + \langle \nabla f(\vec{x}), \vec{y}-\vec{x}\rangle$, $\forall \vec{x},\vec{y}\in K$.

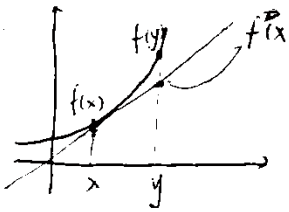


Figure. A 1-dimensional example of the first-order condition.

Second-order condition for convexity: If the function f is twice differentiable, then f is convex iff. $H_f(\vec{x})$ is positive semi-definite (PSD, 半正定). $H_f(\vec{x})_{ij} = \frac{\partial^2 f(x)}{\partial x_i \partial y_j}$

Definition (Lipschitz continuity) A function $f: K\to\mathbb{R}$ is called G-Lipschitz (continuous) with respect to the norm $\|\ \|$ if $|f(\vec{x})-f(\vec{y})| \le G\|\vec{x}-\vec{y}\|$, $\forall \vec{x},\vec{y}\in K$.
例如. 对于 2-范数. 即要 $\|\nabla f(\vec{x})\|_2 \le G$, $\forall \vec{x}\in K$.

$\min\{f(\vec{x}) : c^T\vec{x}+b\}$, $\vec{c},\vec{x}\in\mathbb{R}^n$, $b\in\mathbb{R}$ (linear function).
$\vec{x}_0$, $\vec{u}$, $\|\vec{u}\|_2=1$. $f(\vec{x}_0+\vec{u}) = \vec{c}^T\vec{x}_0 + b + \vec{c}^T\vec{u} = \|\vec{c}\|_2\|\vec{u}\|_2\cos\theta$
minimum when this factor $=-1$.
i.e., opposite direction of the gradient.

Basic GD:
init: $\vec{x}_1 \leftarrow$ any point
for $t = 1, 2, \dots T$,
$\quad \vec{x}_{t+1} \leftarrow \vec{x}_t = \eta\nabla f(\vec{x}_t)$
return $\vec{x} = \frac{1}{T}\sum_{t=1}^{T}\vec{x}_t$

(任意). 可以替近似

<u>Proposition 1</u> Let $f: \mathbb{R}^n \to \mathbb{R}$ be convex, differentiable and G-Lipschitz. $D = \|\vec{x}_1 - \vec{x}^*\|_2$, T
and $\eta = \frac{D}{G\sqrt{T}}$. then the solution $\vec{\bar{x}}$ satisfies $f(\vec{\bar{x}}) \leq f(\vec{x}^*) + \varepsilon$, $\forall \vec{x}^* \in K$.

<u>Theorem 2</u>. (for proving prop 1). Let $f: \mathbb{R}^n \to \mathbb{R}$ be convex, differentiable and G-Lipschitz
The GD algorithm ensures

$$\sum_{t=1}^{T} f(\vec{x}_t) \leq T f(\vec{x}^*) + \underbrace{\overbrace{\frac{1}{2}\eta TG^2}^{\text{like } \varepsilon T \text{ in MWU}} + \overbrace{\frac{1}{2\eta} D^2}^{\frac{\ln N}{\varepsilon} \text{ in MWU}}}_{\text{regret}}$$

Assume Theorem 2 stands, Proof (proposition 1).

$$f(\vec{\bar{x}}) = f\left(\frac{1}{T}\sum_{t=1}^{T} \vec{x}_t\right) \leq \frac{1}{T}\sum_{t=1}^{T} f(\vec{x}_t) \quad \{\text{convex}\}$$

$$\leq f(\vec{x}^*) + \frac{1}{2}\eta G^2 + \frac{1}{2\eta T} D^2 \quad \{\text{Theorem 2}\}$$

$$= f(\vec{x}^*) + \frac{LG}{\sqrt{T}} = f(\vec{x}^*) + \varepsilon.$$

Proof (Theorem 2). $\Phi_t = \frac{\|\vec{x}_t - \vec{x}^*\|_2^2}{2\eta}$

Lemma 3. $\Phi_{t+1} - \Phi_t \leq \langle \nabla f(\vec{x}_t), \vec{x}^* - \vec{x}_t \rangle + \frac{\eta}{2}\|\nabla f(\vec{x}_t)\|_2^2$

Proof (Lemma 3). $\|\vec{a} + \vec{b}\|_2^2 = \|\vec{a}\|_2^2 + \|\vec{b}\|_2^2 + 2\langle \vec{a}, \vec{b} \rangle$

$\vec{a} = \vec{x}_t - \vec{x}^*$, $\vec{b} = \vec{x}_{t+1} - \vec{x}_t = -\eta \nabla f(\vec{x}_t)$

$\Phi_{t+1} - \Phi_t$ ~~$= \frac{1}{2\eta}(\|\vec{x}_{t+1} - \vec{x}^*\|_2^2 - \|\vec{x}_t - \vec{x}^*\|_2^2) = \frac{1}{2\eta}(\|\vec{a}+\vec{b}\|_2^2 - \|\vec{a}\|_2^2) = \frac{1}{2\eta}(2\langle \vec{a}, \vec{b}\rangle + \|\vec{b}\|_2^2)$~~

Lemma 4 $f(\vec{x}_t) + (\Phi_{t+1} - \Phi_t) \leq f(\vec{x}^*) + \frac{1}{2}\eta G^2$

$$\Phi_{t+1} - \Phi_t = \frac{1}{2}\eta\|\nabla f(\vec{x}_t)\|_2^2 + \langle \vec{x}^* - \vec{x}_t, \nabla f(\vec{x}_t) \rangle$$

$$\leq \frac{1}{2}\eta G^2 + f(\vec{x}^*) - f(\vec{x}_t)$$

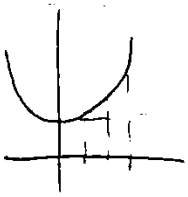
Summing up Lemma 4 for t=1.

$$\sum_{t=1}^{T} f(\vec{x}_t) + \underbrace{\sum_{t=0}^{T}(\Phi_{t+1} - \Phi_t)}_{\Phi_T - \Phi_0} \leq \frac{1}{2}\eta G^2 T + T f(\vec{x}^*)$$

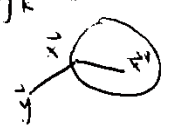
$$\sum_{t=1}^{T} f(\vec{x}_t) \leq \frac{1}{2}\eta G^2 T + T f(\vec{x}^*) + \overline{\Phi_0 - \Phi_T}$$

$$\frac{D^2}{2\eta} \geq 0.$$

Convex function $f: K \to \mathbb{R}$, $K \subset \mathbb{R}^n$. get a minimizer iff. $\langle \nabla f(\vec{x}^*), \vec{y} - \vec{x}^* \rangle \geq 0$, $\forall \vec{y} \in K$.



$\text{proj}_K : \mathbb{R}^n \to K$ $\quad \text{proj}_K(\vec{y}) = \arg\min_{\vec{x} \in K} \|\vec{x} - \vec{y}\|_2$



$\vec{y} - \vec{x} - \vec{x}^*$: obtuse angle.

Fact 5. If $\vec{x} = \text{proj}_K(\vec{y})$ and $\vec{x}^* \in K$, then $\|\vec{x} - \vec{x}^*\|_2 \leq \|\vec{y} - \vec{x}^*\|_2$

...

$\vec{x}'_{t+1} \leftarrow \vec{x}_t - \eta \nabla f(\vec{x}_t)$

$\vec{x}_{t+1} \leftarrow \text{proj}_K(\vec{x}'_{t+1})$

return $\vec{\bar{x}} = \frac{1}{T} \sum_{t=1}^{T} \vec{x}_t$.

$\phi_{t+1} - \phi_t = \frac{1}{2\eta} (\|\vec{x}_{t+1} - \vec{x}^*\|_2^2 - \|\vec{x}_t - \vec{x}^*\|_2^2)$

$= \text{proj}_K(\vec{x}_{t+1}) \leq \|\vec{x}'_{t+1} - \vec{x}^*\|_2^2$

Online GD. function changes over time. (as long as f_t (every) convex)

...

$\vec{x}_{t+1} \leftarrow \vec{x}_t - \eta \nabla f_t(\vec{x}_t)$

...

$f(\vec{\bar{x}}) = f(\frac{1}{T} \sum x_t)$

$\leq \frac{1}{T} \sum f(x_t)$.

Definition (strongly convex - Not too fast) A function $f: K \subseteq \mathbb{R} \to \mathbb{R}$ is **α-strongly convex** if $\forall \vec{x}, \vec{y} \in K$, any of the following holds

(1) Zeroth order. $f(\lambda \vec{x} + (1-\lambda)\vec{y}) \leq \lambda f(\vec{x}) + (1-\lambda) f(\vec{y}) - \frac{\alpha}{2} \lambda (1-\lambda) \|\vec{x} - \vec{y}\|_2^2$, $\forall \lambda \in [0,1]$

(2) First order. If f is differentiable, then $f(\vec{y}) \geq f(\vec{x}) + \langle \nabla f(\vec{x}), \vec{y} - \vec{x} \rangle + \frac{\alpha}{2} \|\vec{x} - \vec{y}\|_2$

(3) Second order. If f is twice-differentiable, then all eigenvalues of $H_f(\vec{x}) \geq \alpha$.

Definition (Lipschitz Smoothness - Not too convex). **β-smooth** if any

(1) Zeroth order. $f(\lambda \vec{x} + (1-\lambda)\vec{y}) \geq \lambda f(\vec{x}) + (1-\lambda) f(\vec{y}) - \frac{\beta}{2} \lambda (1-\lambda) \|\vec{x} - \vec{y}\|_2^2$, $\forall \lambda \in [0,1]$

(2) First order. f differentiable, then $f(\vec{y}) \leq f(\vec{x}) + \langle \nabla f(\vec{x}), \vec{y} - \vec{x} \rangle + \frac{\beta}{2} \|\vec{x} - \vec{y}\|_2^2$

(3) Second order. twice ~~dif~~ differentiable, then all eigenvalues of $H_f(\vec{x}) \leq \beta$.

ε-closeness $T = \Theta(\log(\frac{1}{\varepsilon}))$, $f(\vec{y}) \geq f(\vec{x}) + \langle \vec{z}, \vec{y} - \vec{x} \rangle$