lec1,2,3

Stable Matching: while there is somebody not engaged, let A be an arbitrarily single boy, X be the first girl A has not proposed yet. If X is single, match A and X. Compare current couple: better then change, worse then keep. running time $O(n^2)$.
Random Instance: each boy's list is a random permutation of girls.

$$\mathbb{E}[\#\text{iterations}] = \sum_I \Pr[I]T[I] \le O(n \log n)$$

Intuition: "A boy propose to a random girl he has proposed" is better than "A boy propose a random girl".
Note if every one is engaged then the process is over, transform into "throw ball uniformly into a bin", $O(n \log n)$.
DFS White Path Theorem: consider the time when DFS discover u,v is a descendant of u, iff $\exists$ a while path from $u \to v$(searched points are while, points in queue are grey, unexplored points are black).
SCC strongly connecting components. A directed graph can be partitioned into disjoint SCC, after contracting each SCC into one point, the graph becomes an DAG.
Kosaraju

1. DFS(G), compute enter time($u.f$) and exit time(v,f)
2. compute G^T (reverse all edge)
3. DFS(G^T), consider the adjacent ndoes, they form SCCs

Lemma A: the node with largest $v.f$ belongs to a source comp in G.
Shortest Path From s to t, weight ≥ 0.
Dijkstra: every time choose the minimum cost in for all "to be explored nodes", correctness is gueranteed by: the least cost can not be optimized by any other path.
MST: Kruskal's Algo: sort all edges in the order of weight, if create a cycle, discard. Running time $O(|E| \log |V|)$.
Huffman Codes frequency f_x,encoding length of r: $len(r) = \sum_{x\in S} f_x|r(x)|$, entropy $H = -\sum_x f_x \log f_x$

lec4,5,6 DP and NPC

Tree Decomposition Each node $t \in T$ corresponds to a bag of nodes in the original graph, $V_t \subseteq V$. Each node and edge belongs to some bag, $t_1, t_2, t_3 \in T$ and t_2 lies on the path from t_1 to t_3, if $v \in V_{t_1} \cap V_{t_3}$, then $v \in V_{t_2}$.
Treewidth of G: $\min_T \max_{t\in T} |V_t| - 1$.
Lemma1: Suppose $T - t$ has components $T_1, T_2, \cdots, T_d$, then $G_{T_1} - V_t, \cdots$ has no nodes in common.
Lemma2: Suppose $t_1, t_2 \in T$ are two adjacent bags, then deleting $V_{t_1} \cap V_{t_2}$, disconnect G into ≥ 2 components.
Find max Independent set for graphs with $Tw(G) = O(1)$. find $\min tw$ is NP-hard, if $tw(G) = O(1)$, there is a $2^{O(tw)}n$ algorithm to find the tree decomposition with $\min tw$.
NPC Problems: 3-SAT, IS(Independent Set), HC(Directed Hamilton Cycle), VC(Vertex Cover), SC(Set Cover), 3DM(3-dimensional Matching), Subset-Sum

lec7,8 Approx

FPT: running time $O(poly(n) \times f(k))$, k is the parameter. k can be chosen arbitrarily, but we often choose k to be the size of the solution.
Vertex Cover ob1: $e = (u, v)$ is an edge. $VC(G) \le k$ iff $VC(G-u) \le k-1$ or $VC(G-v) \le k-1$.

$$T(n,k) \le 2T(n-1,k-1) + c^2 \cdot n \cdot k$$

LP relaxation(ILP to LP): $\min w_i x_i$, $x_i + x_j \ge 1 \forall (i,j) \in E$,$OPT(VC) = OPT(ILP) \ge OPT(LPR)$
rounding $\overline{x_i} = \begin{cases} 0, \tilde{x_i} < 0.5 \\ 1, \tilde{x_i} \ge 0.5 \end{cases}$ is a 2 approximation.
LP rounding for SC($O(\log n)$): LPR:$\min \sum_i x_i w_i$,$\sum_{i:e\in S_i} x_i \ge 1$,$\forall e \in U, x_i \in [0,1]$

$$\Pr[e \text{ is not covered}] = \prod_{i=1}^{k} (1 - \tilde{x_i}) \le \frac{1}{e}$$

$$\Pr\{\exists e \text{ is not covered in } O(\log n)\} \le \frac{1}{n}$$

$$\begin{aligned} \mathbb{E}[SOL] &\le 2 \log n \mathbb{E}[1 \text{ round cost}] \\ &\le 2 \log n \cdot OPT \end{aligned}$$

Approximation Ratio: $\alpha(G) = \frac{OPT(G)}{ALG(G)}$.
Load Balancing: n jobs (job j has processing time t_j, m machines $M_1, \ldots, M_m$. Goal: minimize $\max_j T_j$) NP-hard. Solution: sort with $\frac{m}{c}$, this gives approximation ratio 1.5.
k-center problem

$$\begin{cases} d(u,v) \ge 0 \\ d(u,v) = d(v,u) \\ d(u,v) \le d(u,w) + d(w,v) \end{cases}$$

minimize $\max_{v\in V} d(v, C), d(v, C) = \min_{c\in C} d(v, c)$
Algo: guess a r, $S = G$, $C = \emptyset$, while $S \ne \emptyset$, choose arbitrarily $v \in S$, $C = C + v$, delete all node $d(u, v) \le 2r$, if $|c| \le k$, succeed, otherwise guess a larger r.
Claim: if our guess is correct ($r \ge$ OPT), then each node at least delete one optimal cluster.
Knapsack n items, item i has weight w_i, value v_i, knapsack capacity W_i. Goal: max total value. $\overline{v_i} = \left\lceil \frac{v_i}{b} \right\rceil \cdot b$, $\hat{v_i} = \left\lceil \frac{v_i}{b} \right\rceil$, $b = \frac{\epsilon}{2n} \max_i v_i$. Different $\hat{v_i}$ has $\frac{2n}{\epsilon}$.
Algo: $OPT(i, V)$: smallest weight can obtain $\{1, 2, \cdots, i\}$ with total value $\ge V$.

$$OPT(i,V) = \min \begin{cases} OPT(i-1,V) \\ w_i + OPT(i-1, \\ \max(0, V - \hat{v_i})) + w_i \end{cases}$$

$S$is SOL,S^* is optimal.
Thm:$(1+\epsilon)\sum_{i\in S} v_i \ge \sum_{i\in S^*} v_i$

$$\begin{aligned} \sum_{i\in S^*} v_i \le b \sum_{i\in S^*} \hat{v_i} \le b \sum_{i\in S} \hat{v_i} \le \sum_{i\in S} (v_i + b) \\ \le \sum_{i\in S} v_i + nb = \sum_{i\in S} v_i + \frac{\epsilon}{2} \max_i v_i \end{aligned}$$

set cover: U a set of elements. $S_1, \cdots, S_m \subseteq U$ are subsets, w_i weights. Goal: $\min \sum_{i\in C} w_i$.

$$S_i = \frac{w_i}{\#\text{elements covered by } S_i}$$

$S_{i_1}, S_{i_2}, \cdots$ are by greedy.#elements newly covered:$m_1, m_2, \cdots$.#elements remained:$n_1, n_2, \cdots$.$n_1 = n - m_1, n_2 = n - m_1 - m_2$. Claim : $\frac{w_{i_{l+1}}}{m_{t+1}} \le \frac{\text{OPT}}{n_t}$

$$\frac{w_{i_{t+1}}}{m_{t+1}} \le \frac{w(SO_i)}{m(SO_i)}, \forall i = 1, 2, \cdots, p$$

$$\frac{w_{i_{t+1}}}{m_{t+1}} \le \frac{\sum_{i=1}^{p} w(SO_i)}{\sum_{i=1}^{p} m(SO_i)} \le \frac{\text{OPT}}{n_t}$$

$$\text{SOL} = \sum_{i=1}^{k} \frac{\text{OPT}}{n_{t-1}} \cdot m_t \le \text{OPT} \cdot H_n$$

Linear Program min/max $c^T x$, $Ax \le b$
Simplex Algorithm(Danzig): worst case exponential time. Ellipsoid: weak poly.
Max-cut Problem 2-approximation, put into S and S^* with $\frac{1}{2}$, derandomize by pick larger one for each vertex.
TSP double the MST(minimal spanning tree), traverse with a shortcut.

lec9 Div and Conq

Closest Pair find the closest pair of points in n points, $O(n \log n)$.
Divide and Conquer: left side is d_1, right side is d_2, how to find d. set $\delta = \min d_1, d_2$. Divide the area into cells, $\delta \times \delta$, each cell has ≤ 1 points. Possible least pair among two section is ≤ 11, $O(n)$ time to verify.
hash solution: randomly order $P_1, P_2, \cdots, P_n$, start with $\delta = d(P_1, P_2)$, when process P_i, want to find $j < i$,$d(P_i, P_j) < \delta$. Divide the plain into cells of size $\frac{\delta}{2}$. Query 25 neighbor cells in HT. Also note that P_i causing δ change has a low probability. Total running time is $O(n)$.
Pivot Selection Divide into $n/5$ groups, find median of each group, find median of medians, find the rank of the median of medians, divide into two parts, recursively find the element. $T(n) = T(n/5) + T(7n/10) + O(n)$, $O(n)$ time.
FFT $O(n \log n)$algorithm for polynomial multiplication.

$$A(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}$$

$$B(x) = b_0 + b_1 x + \cdots + b_{n-1} x^{n-1}$$

$$C(x) = A(x)B(x)$$

choose $x_j = e^{\frac{2\pi j i}{2n}}$, divide and conquer $A(x) = A_{\text{even}}(x^2) + x A_{\text{odd}}(x^2)$.

lec10 Net Flow

For any cut $C = (C_s, C_t)$,$v(C) = \sum_{u\in C_s, v\in C_t} C_{u,v}$. $v(f) \le V(C)$ for any cut C.
THM:max flow=min cut.

$$\max_{s-t \text{ flow } f} v(f) = \min_{\text{cut } C} V(C)$$

Hall's Theorem: bipartite graph $G = (A, B, E)$, $\forall S \subseteq A$, #neighbors of $S \ge \#S$, then there is a matching that covers A.
Residual graph: for each edge, construct another side, add together is the capacity.
Ford Fulkerson: C_e are integers, running time $O(|E| \cdot \sum_e C_e)$.
f is a flow, (A, B) is a cut, $v(f) = f^{\text{out}}(A) - f^{\text{in}}(A)$.
Image Seg $Q(A, B) = \sum_{i\in A} a_i + \sum_{i\in B} b_i - \sum_{i,j \text{ seperate}} P_{i,j}$, our goal is to maximize Q, change to min-cut cut $C(s,t) = \sum_{i\in A} b_i + \sum_{i\in B} a_i + \sum_{i,j \text{ seperate}} P_{i,j}$.
Baseball Elimination. Claim: Z is eliminated iff $\sum_{x\in T} W_x + \sum_{x,y\in T} g_{x,y} > m|T|$.
THM: Z is eliminated iff max flow $< g_* = \sum g_{u,v}$
Parametric flow: input $c_\lambda(s, v)$ nondecreasing in λ, $c_\lambda(v, t)$ nonincreasing in λ, other linear in λ. Max flowpiecewise linear & concave, compute in $O(nm \log \frac{n^2}{m})$ time.
Disjoint Path: max number of edge/vertex disjoint paths from s to t. If edge, just max flow, if vertex, split each vertex into 2, add a edge with capacity 1.
Circulation with demand and lower bound demand d_v for each $v \in V$, $f^+(v) - f^-(v) = d_v$. Lower bound l_e for each $e \in E$, $l_e \le f_e \le u_e$, find a feasible circulation, turn the lower bound into d_v constraints, $C'_e = C_e - l_e, d'_v = d_v - (\sum_{e\in\delta^+(v)} l_e - \sum_{e\in\delta^-(v)} l_e)$
Project Selection set of project, each profit P_i, a set of precedence constraints, find a set A such that profit(A) is maximized. profit(A)= $\sum_{p_i>0, i\in A} p_i + \sum_{p_i<0, i\notin A} p_i$+const
Densest Subgraph find a subgraph S with $\frac{|E(S)|}{|S|}$ is maximized, guess result is δ, check if $|E(S)| - \delta|S| > 0$ by computing max flow of $S \to_1 E \to_\infty V \to_\delta T$.
Min-cost Flow, Min-cost Matching: given flow F, min cost $C(F) = \sum_{e\in E} c_e F_e$, expand min cost residual graph when find F, strongly poly-time.

lec11 hash

universal hash a class $\mathbb{H}$ of function $h : U \to \{0, 1, \cdots, n-1\}$ is universal if $\forall u, v \in U, u \ne v, \Pr_{h\in\mathbb{H}}[h(u) = h(v)] \le \frac{1}{n}$.
for $p = n$ be prime, for $x \in U$ write $x = \{x_1, x_2, \cdots, x_r\}$,

$$\mathbb{H} = \left\{ h_a(x) = \sum_{i=1}^{r} a_i x_i(p) \right\}$$

is universal.
Perfect hash $O(n)$ space $O(1)$ worst case query time $|S| = n$,$|T| = O(n)$, use two level hash table. $B_i = \{x|h(x) = i\}$, then $\mathbb{E}[\sum_{i=1}^n |B_i|^2] = O(n)$

lec12 Local Search

Metropolis Algorithm generate neighbor x' of x, if $E(x') < E(x)$, accept x', otherwise accept x' with probability $e^{-\frac{E(x')-E(x)}{kT}}$.
THM: stationary distribution π is Gibbs-Boltzman $\Pr_\pi[S] = \frac{1}{Z_T} e^{-\frac{E(S)}{kT}}$

lec13 Stream

 # distinct elements F_0 total elements. maintain $t = O(\frac{1}{\epsilon^2})$ smallest hash value. Let v be the $t-$th value. estimation $\tilde{F_0} = \frac{t}{v}$
main lemma:$\Pr[\tilde{F_0} < (1-\epsilon)F_0] < \frac{1}{100}$ for ϵ small enough. let $X_i = \begin{cases} 1, h(x_i) \le \frac{t}{F_0(1-\epsilon)} \\ 0, \text{otherwise} \end{cases}$,$Y = \sum_{i=1}^n X_i$, $Y < t$, $v > \frac{t}{(1-\epsilon)F_0}$.

$$\Pr[Y < t] \le \Pr[|Y - \mathbb{E}[Y]| > \epsilon t] \le \frac{Var[Y]}{\epsilon^2 t^2}$$

$$\le \frac{\frac{t}{1-\epsilon}}{\epsilon^2 t^2} = O(\frac{1}{\epsilon^2 t}) < \frac{1}{100}$$

Bloom Filter: bit array of m bits, with k hash functions, map elements in U. Adding a element: set k bits at all these positions to 1. Query: if has a 0, not in. If all 1, maybe in.

lec14 Shapley Network

edge cost C_e,$e \in E$,k players, i find a policy P_i to find a path from s_i to t_i. Strategy profile $P = (P_1, \cdots, P_k)$. $C_i = \sum_{e\in P_i} \frac{C_e}{\#\text{players uses } e}$, min total cost $C(P) = \sum_{i=1}^k C_i$ is NP-hard.
NE(Nash Equilibrium): $\forall i$, $C_i(P_i, P_{-i}) \le C_i(P'_i, P_{-i})$. THM: there exists a game social cost of unique NE is $\Theta(\log k)$ times the social optimal.
Price of Stability: PoS=$\frac{C(\text{best NE})}{C(\text{social optimal})}$, PoA=$\frac{C(\text{worst NE})}{C(\text{social optimal})}$
$\Phi(P) = \sum_{e\in E} C_e H(X_e)$ is potential function, where $H(n) = \sum_{j=1}^n \frac{1}{j}$.
lemma: if player i update from P_i to P'_i, then $C_i(P_i, P_{-i}) - C_i(P'_i, P_{-i}) = \Phi(P_i, P_{-i}) - \Phi(P'_i, P_{-i})$.
social optimal= $P^0 \to P^1 \to \cdots \to P^t = NE$,$H_k \cdot C(P^0) \ge \Phi(P^0) > \cdots > \Phi(P^T)$,$Pos = \frac{C(NE)}{C(\text{social optimal})} \le H_k$.
THM: PoS$\le$ $O(\log k)$ for any Shapley network.
Machine Scheduling m machines, n jobs, $P_{i,j}$ time j for machine i.
LP Relaxation: min t,$\sum_j P_{i,j} X_{i,j} \le t, \forall$machine i.$\sum_i X_{i,j} \ge 1$, $\forall$job j, if $P_{i,j} > t$, $X_{i,j} = 0$.
Integrity gap=$\frac{\text{cost of } OPT(I)}{\text{cost of } LP(I)}$, m in this problem.
Draw a bipartite graph with n jobs on the left and m machines on the right, connect j to i if $X_{ij} > 0$, only $m + n$ nonzero constraints, so $m + n$ edges in the graph, almost tree.
Assign j' is i's parent

$$\text{load of } m/c \le \sum_{j:\text{child of } i} P_{i,j} + P_{i,j'}$$

$$= \sum_j P_{i,j} X_{i,j} + P_{i,j'} \le 2T$$

1.5 approximation hard.
Mixing Markov P: transition matrix. N:# states. q^0: initial distribution. $q^t = q^0 P^t$. π: stationary distribution. $\pi P = \pi$. $d_{TV}(a, b) = \frac{1}{2}\|a - b\|_1$. Mixing time $\tau(\epsilon) = \sup \min\{t | d_{TV}(q^t, \pi) \le \epsilon\}$.
THM:$\tau(\epsilon) \le O(\frac{\log N + \log \frac{1}{\epsilon}}{1-\lambda_{\max}})$. Suppose $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_n$,$\lambda_{\max} = \max\{|\lambda_2|, |\lambda_n|\}$.

$$\|q^t - \pi\|_1 \le \sqrt{N} \lambda_{\max}^t \|q^0\|_1$$

define $\pi(S) = \sum_{x\in S} \pi(x)$

$$\Phi(S) = \frac{\sum_{x\in S, y\in \bar{S}} \pi_x P_{xy}}{\min\{\pi(S), \pi(\bar{S})\}}$$

normalized conductance $\Phi = \min_S \Phi(S)$, $\frac{\delta}{2} \le \Phi \le \sqrt{2\delta}$, where δ is eigengap

$$\frac{c \log \frac{1}{\epsilon}}{\Phi} \le t_{\text{mix}} \le \frac{c \log \frac{1}{\pi^* \epsilon}}{\Phi^2}$$

lec15

TUM: a matrix A is called TUM if the determinant if every square submatrix is $\{0, 1, -1\}$.
HK THM: A is TUM iff for any integral vector b, $P = \{x | Ax \le b\}$ is an integral polyhydrogen.
Prop: A is TUM, then for every intertible submatrix U, U^{-1} is integral.
GH THM: $A_{m\times n}$ is TUM iff for any subset $R \in [m]$, there is a partition $R = R_1 \cup R_2$, $R_1 \cap R_2 = \emptyset$

$$\left(\sum_{i\in R_1} a_{i,j} - \sum_{i\in R_2} a_{i,j} \right) \in \{0, \pm 1\}, \forall j \in [n]$$

A network matrix is TUM: consecutive 1 matrix is network matrix (row direction has its 1 continuous).
Online Algorithm

$$\text{competitive ratio} = \frac{\text{SOL of online algorithm}}{\text{OPT of offline algorithm}}$$

Ineqs

Chernoff Bound Independent $X_i \in [0, 1]$, $X = \sum_{i=1}^n X_i$,$\mu = \mathbb{E}[X]$.

$$\Pr[X > (1+\epsilon)\mu] < \left[\frac{e^\mu}{(1+\delta)^{1+\delta}} \right]^\mu \le e^{-\frac{\mu\delta^2}{3}}$$

$$\Pr[X < (1-\epsilon)\mu] < e^{-\frac{\mu\delta^2}{2}}$$

Hoeffding Bound Independent $X_i \in [a_i, b_i]$, $X = \sum_{i=1}^n X_i$,$\mu = \mathbb{E}[X]$,

$$\Pr[X > \mu + \epsilon] < e^{-\frac{2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}}$$

$$\Pr[X < \mu - \epsilon] < e^{-\frac{2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}}$$

Master Theorem for $T(n) = aT(\frac{n}{b}) + f(n)$

$$T(n) = \begin{cases} \Theta(n^{\log_b a}) & f(n) = O(n^{\log_b a - \epsilon}) \\ \Theta(n^{\log_b a} \log n) & f(n) = \Theta(n^{\log_b a}) \\ \Theta(f(n)) & f(n) = \Omega(n^{\log_b a + \epsilon}) \end{cases}$$