- For two positive functions f, g defined for positive ints:
  - $f = O(g)$ iff $\exists N, C$ s.t. $f(n) \leq Cg(n)\ \forall n \geq N$
  - $f = \Omega(g)$ if $g = O(f)$
  - $f = \Theta(g)$ if $f = O(g)$ and $g = O(f)$
- $\lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} = \begin{cases} \text{finite} & \implies f(n) = O(g(n)) \\ \text{non-zero} & \implies f(n) = \Omega(g(n)) \end{cases}$
- $a^{\log_b n} = n^{\log_b a}$
- **Euclid's Algorithm**: if $a \geq b > 0$, then $\gcd(a,b) = \gcd(b, a \mod b)$
- **Fermat's Little Theorem**: If p is prime then for every $a : 1 \leq a < p, a^{p-1} \equiv 1 \mod p$.
- **Euler's Theorem** :If a, n coprime: $a^{\phi(n)} = 1 \mod n$
- RSA
  - Primes p, q; $N = pq$
  - $ed \equiv 1 \mod (p-1)(q-1)$
  - Public key: (N, e). Private key: d
  - Encryption $x \mapsto x^e$. Decryption $y \mapsto y^d$
- H is a universal hash function if for every two items x and y, exactly $|H|/n$ of the functions map x and y to the same bucket (n = number of buckets)
- **Master Theorem**: Given $T(n) = aT(\frac{n}{b}) + O(n^d)$
- $T(n) = \begin{cases} O(n^d) & \text{if } d > \log_b a \\ O(n^d \log n) & \text{if } d = \log_b a \\ O(n^{\log_b a}) & \text{if } d < \log_b a \end{cases}$
- FFT-recursive: $A(\omega^j) = A_e(\omega^{2j}) + \omega^j A_o(\omega^{2j}), \omega = e^{2\pi i/n}$
- FFT-Matrix: $M_n(\omega)_{j,k} = \omega^{jk}$, for $0 \leq j, k < n$. Inverse: $M_n(\omega)^{-1} = \frac{1}{n} M_n(\omega^{-1})$
- After DFS traversal, (u, v) is a:
  - tree edge iff $[\text{pre}(v), \text{post}(v)] \in [\text{pre}(u), \text{post}(u)]$
  - back edge iff $[\text{pre}(u), \text{post}(u)] \in [\text{pre}(v), \text{post}(v)]$
  - cross edge iff $[\text{pre}(v), \text{post}(v)]$ then $[\text{pre}(u), \text{post}(u)]$
- SCC: Vertex with highest post number is in a source component. Can run on G^R to identify sinks.
- Dijkstra /Bellman-Ford for paths. Linearize for DAGs.
- Greedy. MST: Kruskal's, Prim's. Huffman encoding, Horn SAT, set cover approximation.
- **Runtimes**:
  Dijkstra: $O((|V| + |E|) \log |V|)$ with binary heap
  Bellman-Ford: $O(|V||E|)$
  Kruskal: $O(|E| \log |V|)$; if E sorted, then $O(|E| \log^* |V|)$
- DP: Longest common subsequence, etc. Floyd-Warshall for all-pairs shortest paths (expand permissible intermediate nodes)
- LP: Simplex, max-flow, bipartite matching
  - Standard Form: $\min \vec{c} \cdot \vec{x}$, $A\vec{x} \geq \vec{b}, \vec{x} \geq \vec{0}$
  - Dual LP: $\max y^T b$, $y^T A \geq \vec{c}$, $\vec{y} \geq \vec{0}$

Ford-Fulkerson (Runtime: $O(|E| \times F)$)

- Find $s-t$ path, p with positive residual capacity (w/ DFS)
- Let δ be minimum $r(e)$ of edge in p
- For each edge, $e = (u, v)$ in path
  - if e in G, $f(e) = f(e) + \delta$
  - if (v, u) in G, $f(v, u) = f(v, u) - \delta$
- compute residual capacities
- repeat; or terminate if no $s - t$ path with pos. residual capacity

Strategic / Two-Person Zero-Sum Games

- Row ($\vec{x}$) maximizes, column ($\vec{y}$) minimizes.
- Row 1$^{\text{st}}$: $\min_y \max_x (x^T Ay) = \max_x \min_y (x^T Ay)$:Col 1$^{\text{st}}$
- LP for Row: $\max z$, $\sum x_i = 1$. $z \leq$ the expected value for each pure strategy from column (e.g. $z \leq x_2 - x_3$)
- LP for Col: $\min w$, $\sum y_i = 1$. $w \geq$ the expected value for each pure strategy from row (e.g. $w \geq y_1 - y_3$)

Simplex

- Start at origin (assume feasible). If all coefficients $c_i \leq 0$ in objective function (for maximization), then optimal
- Increase x_i with highest positive coefficient c_i in objective function until a constraint is hit
- Repeat with new coord system defining curr pt as origin
  - y_i is distance from constraint i. $y_i = b_i - \vec{a_i} \cdot \vec{x}$ ($\vec{a_i}$ is row i of A, i.e. the coeffs for constraint i)
  - Solve for x_i's from sys of eq, sub into obj/constraints
- Origin not feasible? Use new LP. New vars z_i, subtract from LHS of each constraint. $\min(\sum z_i)$, $z_j \geq 0$. Start at $x_j = 0$, $z_i = \max(-b_i, 0)$. Result x values are new start vertex for original LP.

NP complete problems

- SAT: Boolean formula in conjunctive normal form (CNF): clauses containing OR of literals, AND of these clauses
- TSP: Is there a tour, which visits each node exactly once, within budget?
- RUDRATA PATH: Path passing thru each vertex exactly once
- BALANCED CUT: With budget b for cut, partition into two sets such that each has $\geq 1/3$ of elems
- ILP: Linear programming, but constrain variables to be integer (ZOE: constrain to be binary, $A\vec{x} = \vec{1}$)
- 3D matching: Given m valid tuples, match n boys, girls, and pets: find n disjoint triples that get along
- INDEPENDENT SET: Graph, integer g: find g vertices that are independent (no 2 share edge)
- VERTEX COVER: b vertices that touch every edge
- DOMINATING SET: b vertices such that every edge is in the set, or has a neighbor in the set
- SET COVER: given subsets, select b subsets such that union is the entire set
- CLIQUE: g vertices, each is connected to every other
- LONGEST PATH: is there a simple path of length g or more from s to t?
- KNAPSACK: Given weights and values for items, find best knapsack value with weight at most W

Reductions

- Any problem in NP $\to$ SAT (through CSAT)
- SAT $\to$ 3SAT
- 3SAT $\to$ {Independent Set, 3D Matching}
- Independent Set $\to$ {Vertex Cover, Clique}
- 3D Matching $\to$ ZOE
- ZOE $\to$ {Subset Sum, ILP, Rudrata Cycle}
- Rudrata Cycle $\to$ TSP

Coping with NP Completeness

- Backtracking: Try an assignment, test if it meets our constraints. If not, then reject.
- Branch and Bound: To reject a subproblem in a minimization problem, we must be certain that its cost exceeds that of some other solution.
- Approximation Alg.: For a min. problem, find a solution with factor $\alpha_A = \max\limits_I \frac{A(I)}{OPT(I)}$ away from the optimum.
- Local Search: Suppose we have a set of solutions. We define a neighborhood structure to relate these solutions, then we we seek the local optima.
- Matching - subset of edges that have no vertices in common, any matching is a lower bound on the optimal solution for vertex cover