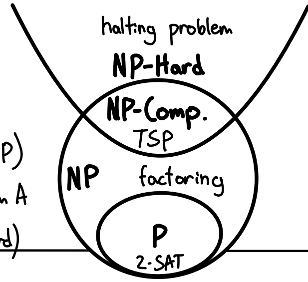
P=NP?



P: solvable in polynomial time

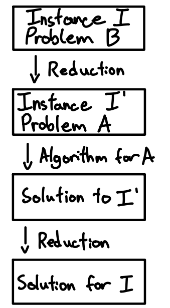
NP: solutions can be verified in polynomial time ('yes solutions')

NP-Hard: at least as hard as the hardest problem in NP (may be outside NP)

- if Problem A is in NP-Hard, every problem in NP can be reduced to Problem A

NP-Complete: problem in NP that all problems can reduce to (NP & NP-Hard)

Reductions



Using a problem that we know how to solve as a subroutine for a different problem

$A \rightarrow B$ implies...

- B is at least as hard as A
- if there is an efficient solver for B, there is an efficient solver for A
- if there is NO efficient solver for A, there is NO efficient solver for B
- a solution to an instance of A exists iff a solution to the corresponding instance of B exists

To prove NP-Complete:

1. Show that the problem is in NP (straight forward)
2. Reduce from the most similar NP-C problem: (1) Φ YES $\Rightarrow$ Φ' YES, (2) Φ' YES $\Rightarrow$ Φ YES

To prove NP-Hard:

1. Reduce from an NP-Hard problem

Coping with NP Completeness

Approximate Algorithms: for an instance I of a minimization problem, approximation ratio $\alpha_A = \max_I \frac{A(I)}{OPT(I)}$

Vertex Cover: Find a maximal matching (disjoint edges) M. return all end points in M. OPT $\geq$ # of edges in M $\Rightarrow \alpha_A = 2$

Metric TSP: Find an MST T, run DFS on T, skip all repeated vertices in traversal. OPT $\geq$ MST $\rightarrow$ 2 OPT $\geq$ A $\Rightarrow \alpha_A = 2$

Randomized Algorithms

Behavior of algorithm changes based on a random string r

Las Vegas:

- Correctness is guaranteed
- Runtime is random
- Goal: Analyze expected worst-case runtime: $T(n) = \max_x [E_r(x, r)]$; show that runtime is low with high probability

Monte Carlo:

- Correctness is random
- Runtime is always bounded
- Goal: Output a number that is close to true value with high probability

- Amplification: $Pr(S) = p$, $Pr(F) = 1-p = q$, $Pr[\text{all } F] = q^k$
  we want $q^k \leq \delta \rightarrow k \ln q \leq \ln \delta \rightarrow k \geq \frac{\ln(1/\delta)}{\ln(1/q)}$
  use $1-x \leq e^{-x} \rightarrow k \geq \frac{1}{p} \ln \frac{1}{\delta}$, $\delta = 0.001 \rightarrow k \geq 6.91/p$
- Union Bound: $Pr[\bigcup_{i=1}^{n} A_i] \leq \sum_{i=1}^{n} A_i$
- $\sum_{k=0}^{n-1} ar^k = \frac{a(1-r^n)}{1-r}$

** $(1-p)^{(10/p)} \leq e^{(-10)} \leq 0.001$

*expected # of events = sum of probabilities of each event

Linearity of Indicators and Indicators: (1) $E[X+Y] = E[X] + E[Y]$, (2) $E[cX] = cE[X]$, (3)* $E[\sum X_i] = \sum Pr[E_i]$

QuickSelect (Las Vegas): Unsorted List $\rightarrow$ Median, pick random pivot, $T(n) \leq cn + \frac{1}{2}T(3n/4) + \frac{1}{2}T(n-1) \rightarrow E[T(n)] = O(n)$

One-Sided Error: YES $\Rightarrow$ YES, NO $\Rightarrow$ NO with probability p, to get Pr(Success) ≥ 0.999, for i in range(10/p){if ALG == NO, ret NO} ret YES **

Two-Sided Error: ALG outputs correct with probability $\frac{1}{2} + \varepsilon$, 0.999 $\rightarrow$ for i in range $(1/\varepsilon^2)$ {record ALG output} ret majority

Primality Testing (Monte Carlo): Pick a random x from $1, \dots, N-1$, check if $x^{N-1} \equiv 1 \pmod{N}$, if so ret PRIME, $C + \cancel{C} \to p \geq 1/2$

Karger's Algorithm (Monte Carlo): Pick a random edge, contract, until 2 vertices left; e left $\geq [(n-i+1) + |C|]/2$, $Pr(C) = 2/(n-i+1)$

- $Pr(\text{Success}) \geq \prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right) = 1 / \binom{n}{2}$

Online Algorithms

Regret: Alg's # of mistakes - OPT (# of mistakes by best expert); $\sum_{t=1}^{T} 1(\text{guess}^{(t)} \neq \text{real}^{(t)}) \lesssim \min_i \sum_{t=1}^{T} 1(o_i^{(t)} \neq \text{real}^{(t)})$

Halving Algorithm: for at least one perfect expert, $T = O(\log n)$, regret $= O(\log n)$

Weighted Majority: penalize incorrect experts $w_i^{(t+1)} \leftarrow w_i^{(t)}(1-\varepsilon)$, else $w_i^{(t)}$, $\varepsilon = 0.5 \to E[r] \leq 1.4\,OPT + 2.4 \log(n)$

Random Weighted Majority: no regret for the right eps, samples expert at random using weighted distribution rather than choosing the expert with the largest weight, experts' behavior is NOT randomized

MW Garuntee: $E[C_{ALG}(T)] \leq \min_i C_i(T) + O(\sqrt{T \log N})$ or $E[\text{Regret}(T)] = E[C_{ALG}(T) - OPT] \leq O(\sqrt{T \log N})$

asymptotics

- $f(n) = O(g(n))$
- $f(n) = \Theta(g(n))$
- $f(n) = \Omega(g(n))$

$\lim_{n\to\infty} \frac{f(n)}{g(n)}$: $< \infty$, $= c$, > 0

exponential > polynomial > logarithm

$3^n > 2^n$ > $n^5 > n^4 > n^{0.1}$ > $(\log n)^3 > (\log n)^2 > \log \log n$

- try substitution! or L'Hopital's!

maths

- $\log_b(M^k) = k \cdot \log_b M$
- $\log_b(b^k) = k$
- $a^{\log_b n} = n^{\log_b a}$
- $\log_b n = \frac{\ln n}{\ln b}$
- $\log_b(M \cdot N) = \log_b M + \log_b N$
- $\log_b(M/N) = \log_b M - \log_b N$
- $1/\log_a b = \log_b a$
- $\sum_{i=1}^{k} \frac{1}{i} = \log k$
- $\ln n \to 1/n$
- $a^n \to a^n \ln n$
- $(\ln n)^k \to k(\ln n)^{k-1}/n$
- $n \log n \to \log n$
- $1 + 2 + \dots + n = O(n^2)$
- $1 + \frac{1}{2} + \dots + \frac{1}{n} = O(\log N)$
- $a + ar + \dots ar^n = \frac{a(1-r^n)}{1-r}$
- $1 + p + p^2 + \dots p^k = \frac{p^{k+1}-1}{p-1}$
- inf geometric $\to \frac{a}{1-r}$

master's theorem

$T(n) = aT(n/b) + O(n^d)$

d	$>$ $\log_b a$	$O(n^d)$
d	$=$ $\log_b a$	$O(n^d \log n)$
d	$<$ $\log_b a$	$O(n^{\log_b a})$

BFS/DFS: $O(|V| + |E|)$

Kosaraju's: $O(|V| + |E|)$

- run DFS
- run DFS on G_{rev} w/ decreasing post-order

Dijkstra's: $O((|E| + |V|) \log |V|)$

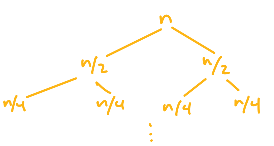
- init dist $= \infty$, par = None
- dist(s) = 0, add to minH
- update all neighbors of first

Bellman-Ford: $O(|V||E|)$

- works w/ neg edges!
- repeat $|V| - 1$ times:
  - for each e in E:
    - update dist(v)

tree method

$T(n) = 2T(\frac{n}{2}) + O(\frac{n}{\log n})$ w/ $T(1) = 1$



$n/\log n$

$\frac{n}{2} \log \frac{n}{2} \cdot 2 = n/\log \frac{n}{2}$

$n/\log \frac{n}{4}$

n/n | $n/\log 1$

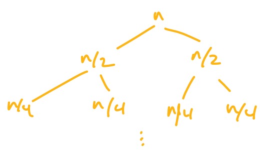
$$n \sum_{j=0}^{\log n} \frac{1}{\log \frac{n}{2^j}} = n \sum_{j=0}^{\log n} \frac{1}{\log n - \log 2^j}$$

$$= n \sum_{j=0}^{\log n} \frac{1}{\log n - j}$$

$$= n \left(\sum_{i=1}^{\log n} \frac{1}{i} \right)$$

$$= n \log \log n$$

$T(n) = 2T(n/2) + O(n \log n)$



$n \log n$

$n \log n/2$

$n \log n/4$

$n \log n/n$

$$n \sum_{i=0}^{\log n} \log \frac{n}{2^i} = n \sum_{i=0}^{\log n} \log n - i$$

$$= n \log^2 n$$

Squeeze + guess & check

- guess U/L bounds

$T(n) = T(3n/5) + T(4n/5)$

guess an^b, $a = 1$ since $T(1) = 1$

$T(n) = n^b = (3n/5)^b + (4n/5)^b$

$\Rightarrow b = 2$

change of variables

$T(n) = 2T(\sqrt{n}) + O(\log n)$

$m = \log n$

$T(2^m) = 2T(2^{m/2}) + O(m)$

$S(m) = T(2^m)$

$S(m) = 2S(m/2) + O(m)$

$\Rightarrow S(m) = m \log m$

$T(2^m) = m \log m$

$T(2^{\log n}) = T(n) = \log n \log \log n$

$T(n) = T(\sqrt{n}) + 1$

$m = \log n$

$T(2^m) = T(2^{m/2}) + 1$

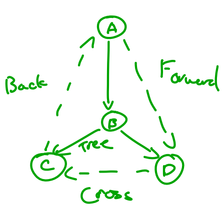
$S(m) = T(2^m)$

$S(m) = S(m/2) + 1$

$\Rightarrow S(m) = \log m$

$T(2^m) = \log m$

$T(2^{\log n}) = T(n) = \log \log n$



Tree/Forward if [[ ]] (u v v u)

pre(u) < pre(v) < post(v) < post(u)

Back if [[ ]] (v u u v)

pre(v) < pre(u) < post(u) < post(v)

cross if: [ ] [ ] (v v u u)

pre(v) < post(v) < pre(u) < post(u)