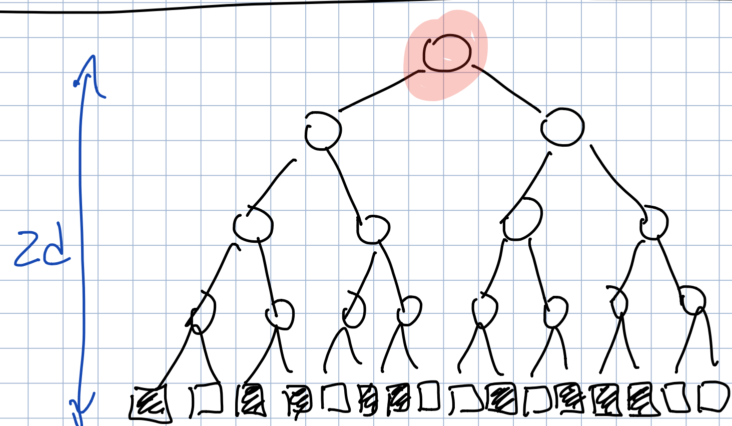
Dynamic Programming
Randomized Algs/DS] MT1
Flows, Cuts, LP] MT2
NP-hardness, Approx Algs



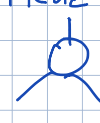
You and Death alternate

□ WIN

▨ DIE

Q: Can you win?

□ = TRUE ▨ = FALSE



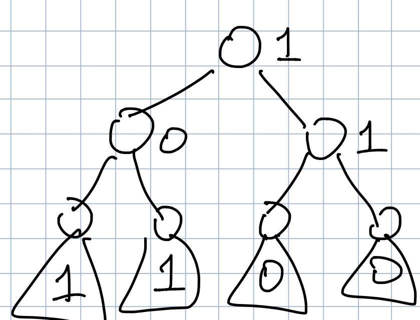
NAND

Time: $O(4^d) = O(n)$

(b) Prove you can't avoid worst-case: every leaf

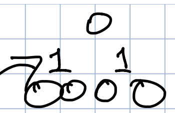
(c) Randomize! $O(c^d)$ exp. time for some $c < 4$.

Postorder, but randomly choose first child at every node

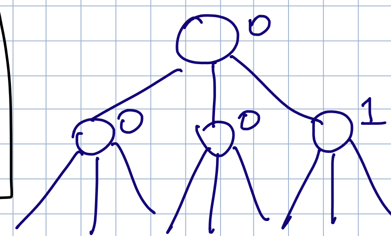


$T(2d) \leq 3 \cdot T(2d-2)$

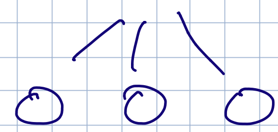
$\Rightarrow \underline{O(3^d)}$ exp time.



$\frac{1}{2}\cdot 1 + \frac{1}{2}\cdot 2 = \frac{3}{2}$



Random Postorder



2 recursive calls

0 0 1 — 2 calls

0 1 0 — 3 calls

1 0 0 — 3 calls

$$E[\#\text{calls}] = \frac{1}{3}\cdot 2 + \frac{1}{3}\cdot 3 + \frac{1}{3}\cdot 3 = \frac{8}{3}$$

$$\Rightarrow T(d) \leq \frac{8}{3}\cdot T(d-1) = O\left(\left(\frac{8}{3}\right)^d\right)$$

burned pancake sorting

Gates + Papadimitriou

7 2 3 4 1 6 5

7 2 5 6 1 4 3

7 2 5 6 3 4 1

$\downarrow$?

1 2 3 4 5 6 7

7 2 3 4 1 6 5

7 2 3 4 5 6 1 ?

1 6 5 4 3 2 7

$T(n) \le 3 + T(n-1)$

$\Rightarrow 3n-2$

MaxSum(3, 1000)

$$\begin{bmatrix} 5 & -2 & 13 & 1 \\ 6 & 5 & -4 & 3 \\ 2 & 9 & 7 & 8 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

Input: $A[1..n, 1..4]$

Choose subset with max sum, but forbidding adj. pairs

0000
0001
0010
0100
1000
0101
1010
1001

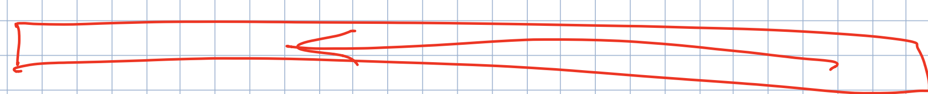
$MaxSum(i, b)$

= Maximum sum from rows $i..n$ where row $i-1$ uses bitpattern b.

$$MaxSum(i,b) = \begin{cases} 0 & \text{if } i > n \\ \max\limits_{b' \text{ consistent with } b} \left(\sum_{j=1}^{4} b'[j] \cdot A[i,j] + MaxSum(i+1, b') \right) \end{cases}$$

Initial call: $MaxSum(1, 0000)$

Data structure: $n \times \cancel{16}$ array
8
$O(1)$



$O(n)$ time □

Bus home from Siebel

List of bus routes: Stops and times $\Big\}$ length N

Siebel home

Minimize time waiting for buses out in the rain

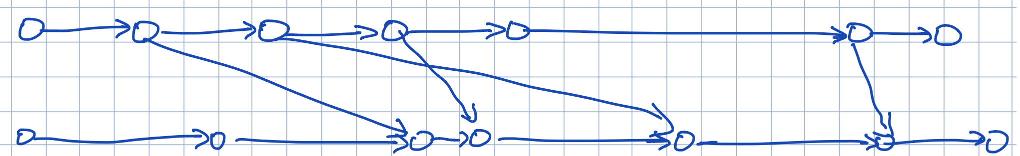
Build a graph:

$V = \{(\text{stop}, \text{time})\}$ $|V| \le N$

$|E| \le 2N$

$E = \{(\text{stop}, \text{time1}) \to (\text{stop}, \text{time2})$ for all successive times$\}$

$\cup \{(\text{stop}_1, \text{time}_1) \to (\text{stop}_2, \text{time}_2)$ consec on same busroute$\}$



$$w(e) = \begin{cases} 0 & \text{if } e \text{ is on a bus} \\ 0 & \text{if stop} = \text{Siebel or home} \\ \text{time}_2 - \text{time}_1 & \text{o/w} \end{cases}$$

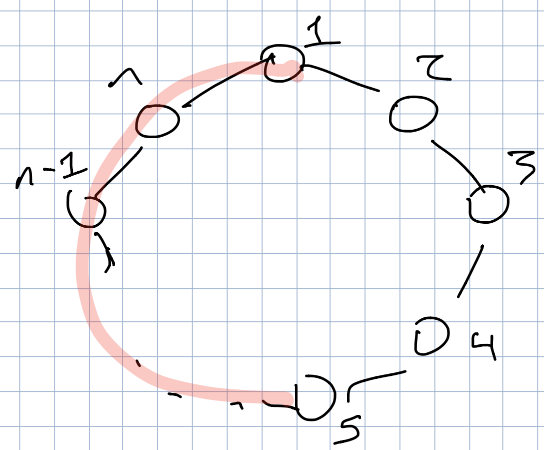
Shortest path from (Siebel, t_{min} now) to (home, ∞ ~~dinner~~ sleep)

Dijkstra $O(E \log V) = O(N \log N)$ time

DAG! DFS/DP $O(E) = O(N)$ time

longest(v) = max length $v \rightsquigarrow t$

$$= \begin{cases} 0 & \text{if } v = t \\ \infty & \text{if } v \ne t \text{ and } v \text{ sink} \\ \max_{v \to w} \ell(v \to w) + \text{longest}(w) \end{cases}$$



Route calls $3 \to 7$

$6 \to 45$

$5 \to 17$

$\vdots$

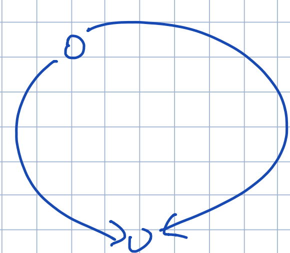
to minimize congestion $\boxed{\max_{e}(\#\text{calls thru } e)}$

Problem: Find a 2-approx algorithm

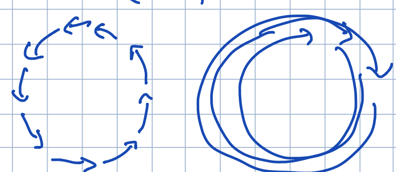
$$A(x) \le f(x) \le 2 \cdot OPT(x)$$

we know $A(x) \ge OPT(x)$

$$OPT(x) \ge \sum_{\text{call } i} \overset{\text{shortest}}{\text{length}}(i) \cdot \frac{1}{n}$$



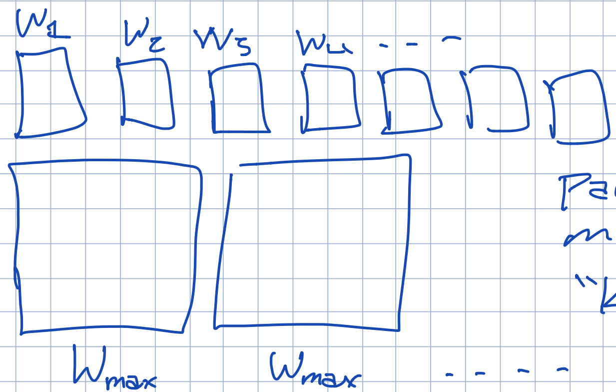
Guess algo: Route ~~Clockwise~~.



two paths together length n

Either cl or ccw, whichever is better

for each call, route shortest to increase as little as possible



Pack boxes into bins
minimize # bins
"Knapsack problem"

2-appx

$$A(x) \le f(x) \le 2 \cdot OPT(x)$$

$$OPT(x) \ge \left\lceil \sum_i w_i / W_{max} \right\rceil$$

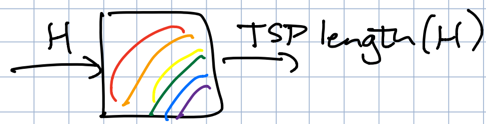
Sort by decreasing weight

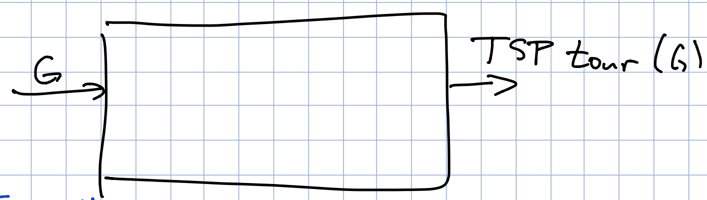
Best fit: for $i \leftarrow 1$ to n
put item i into emptiest nonempty bin
or
add bin if item i doesn't fit...

First fit: $j \leftarrow 1$
for $i \leftarrow 1$ to n
if item i fits
put in j
else
$j \leftarrow j+1$
put i into j

Claim: At most one bin is less than half full.

$$A(x) \le 1 + 2 \sum_i w_i / W_{max} \le \cancel{1+} 2\, OPT(x)$$





For all edges e

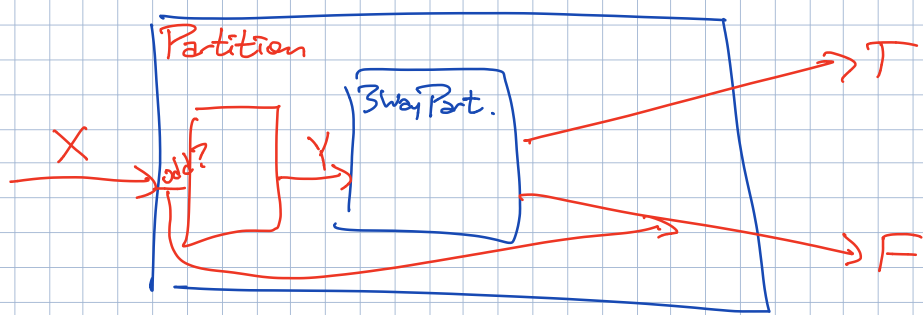
If TSPLength(G) = TSPLength(G − e)

$G \leftarrow G - e$

return G

3WayPartition $X \to A \cup B \cup C$

$\Sigma A = \Sigma B = \Sigma C$



$Y = X \cup \{\Sigma X/2\}$

4 Reduction
6 Proof
↳ = 3 + 3

Proof:

($\Rightarrow$) Suppose X can be partitioned

$X = A \cup B$ where $\Sigma A = \Sigma B$

Then $\Sigma A = \Sigma B = \Sigma X/2$

So Y can be 3-partitioned.

($\Leftarrow$) Suppose Y can be 3-partitioned

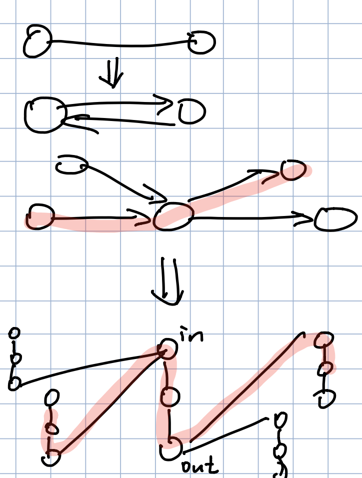
$Y = A \cup B \cup C$

where $\Sigma A = \Sigma B = \Sigma C = \Sigma Y/3$
$= \Sigma X/2$

So wlog $C = \{\Sigma X/2\}$

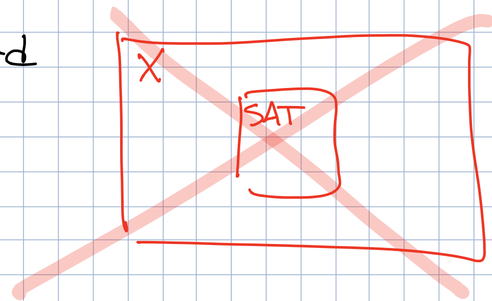
$\Rightarrow A \cup B = X$

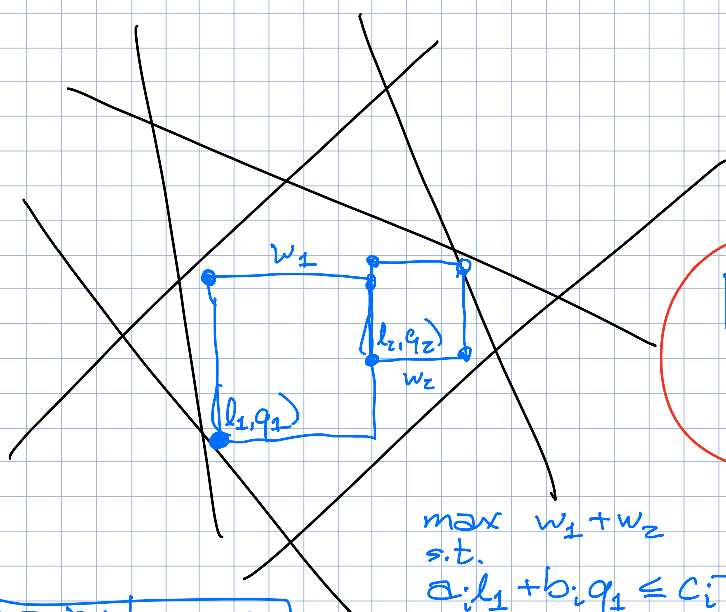
So X can be partitioned. $\square$

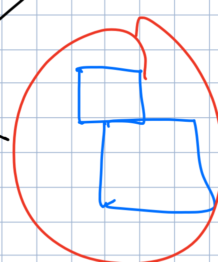


Box Depth → Clique

Prove X is NP hard







$\boxed{a_i x + b_i y \le c_i}$

max $w_1 + w_2$

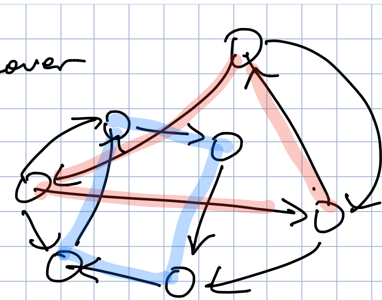
s.t.

$\left.\begin{array}{l} a_i l_1 + b_i q_1 \le c_i \\ \vdots \end{array}\right\} 8n \; \} 8n$

$\boxed{l_2 \ge l_1 + w_1}$

$q_2 \ge q_1 + w_1$

Cycle Cover

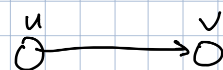


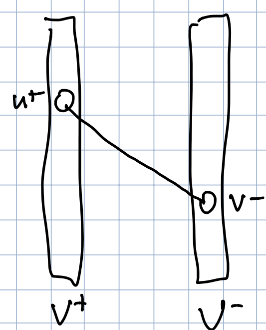
Input:
directed $G=(V,E)$

Output: T/F

Collection of cycles in G touching each vertex once

For every node v
assign a node next(v)





Build bipartite graph H:

$V(H) = V^+ \cup V^-$

$E(H) = \{u^+v^- \mid u \to v \in E\}$

~~Find perfect matching in H.~~

IF H has perfect matching
return T
else
return F

$\boxed{O(V \cdot E) \text{ time}}$