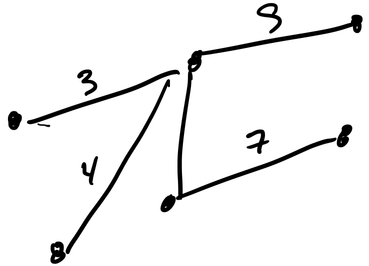
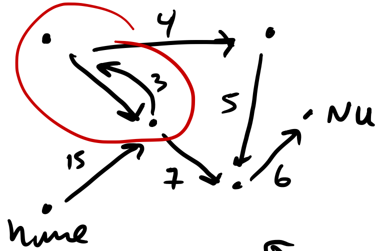
Intro to graphs





Shortest path problem

↖ <u>not</u> a DAG

DAGs – directed acyclic graphs

directed ↑ edges have direction

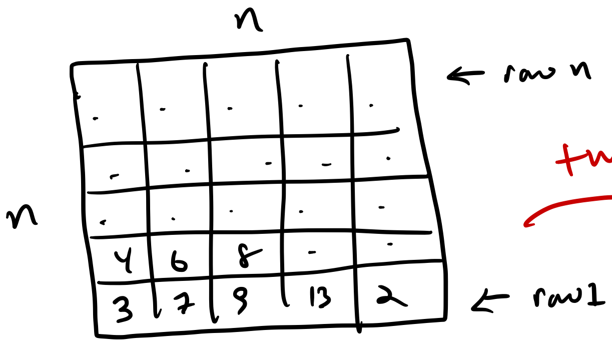
acyclic ↑ no cycles

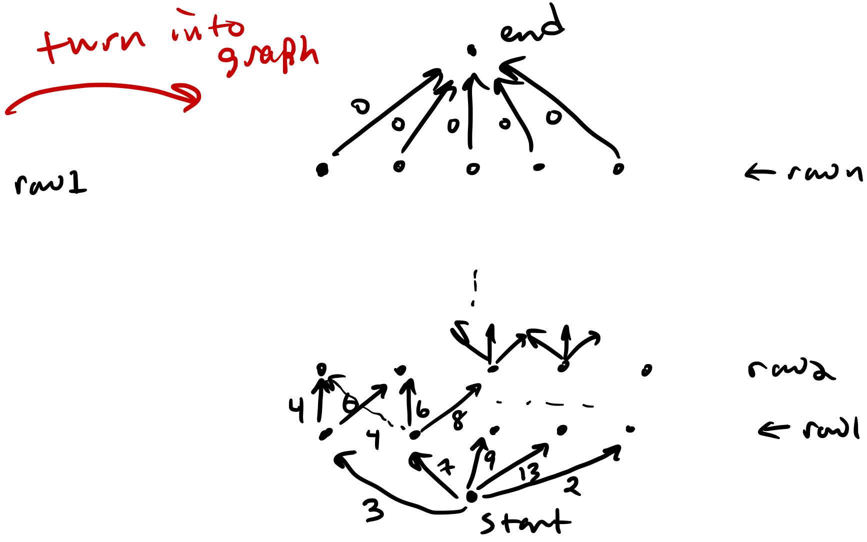
finding s.p. in a DAG

$\Theta(V+E)$ where $|V| = \#$ vertices, $|E| = \#$ edges

① Chess board Problem

Goal: minimize cost





$P(i,j)$ = price or cost of square (i,j)

- step on board at row 1
- move forward <u>or</u> diagonally right <u>or</u> diagonally left
- step off after row n
- pay price associated w/ every square step on

- every <u>solution</u> is a <u>path</u> in <u>graph</u>
- <u>cost</u> of <u>solution</u> is <u>path</u> <u>length</u>
- <u>shortest</u> <u>path</u> ⇒ min cost solution

- size of graph

  $|V| = 2 + n^2 = \Theta(n^2)$

  $|E| = \Theta(n^2)$

  $|V| + |E| = \Theta(n^2)$

- graph is a <u>DAG</u>

- find the s.p. in graph

  in time <u>linear</u> in size of graph, $\Theta(n^2)$, so $\Theta(n^2)$

② Rental Car Problem

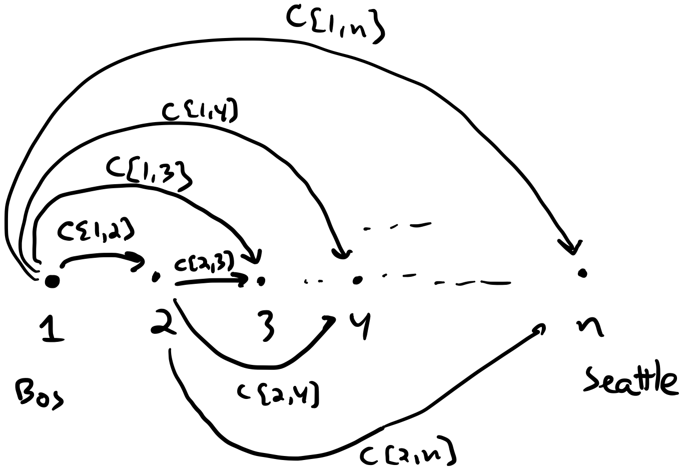
• Boston • toledo • Denver • Bozeman • seattle

Cities $1 \to n$
Bos → Sea

- $C\{i,j\}$ = cost of renting car from $i \to j$

- Goal: find set of rental cars to min. cost and get from Bos → Sea

  - no back tracking



- every soln. to rental car problem is a path in the graph
- Cost of soln is the length of path
- shortest path = min cost soln.

$|V| = \Theta(n)$

$|E| = \Theta(n^2)$

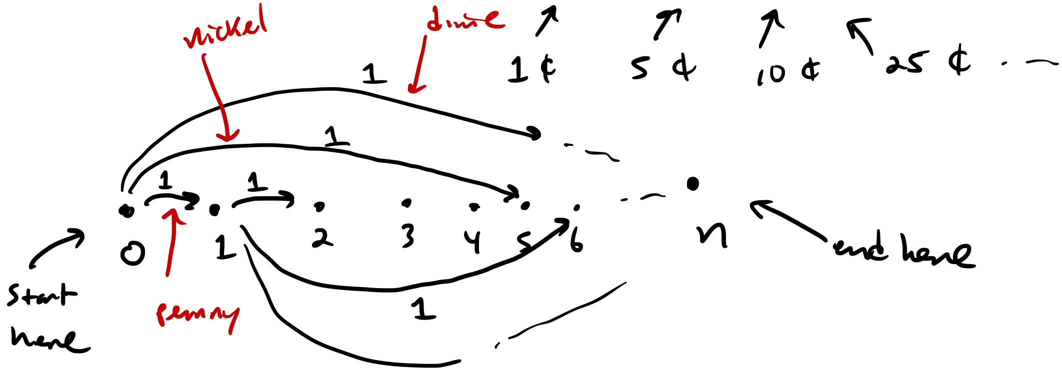
$|V| + |E| = \Theta(n^2)$

- graph is a DAG

S.P. alg. runs in $\Theta(n^2)$

③ Coin changing Problem

- n cents, make change using fewest # coins
- Coins of denominations d[1], d[2], --- d[k]



- every soln to coin changing is a path in graph from 0 to n
- Cost of soln is length of path
- shortest path = min cost soln.
- graph is a <u>DAG</u>

$|V| = \Theta(n)$

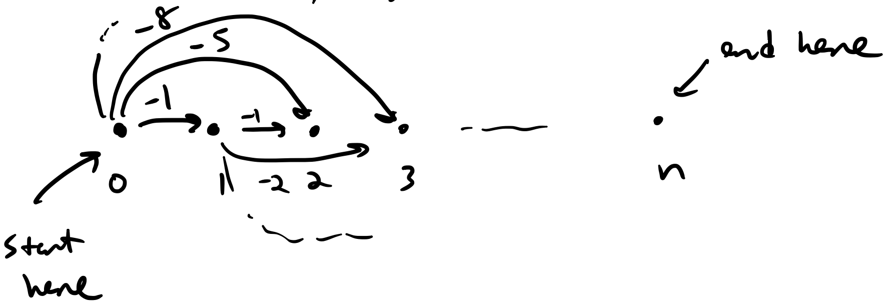
$|E| = \Theta(n \cdot k)$

$|V| + |E| = \Theta(n \cdot k)$

S.P. soln. takes $\Theta(n \cdot k)$

④ Rod cutting

piece length	1, 2, 3, 4, ...
prices	1, 5, 8, 13, ...



- every soln. to rod cutting is a path in graph
- value of soln. is the negative of path length
- shortest (i.e. most negative) path is max value soln.
- graph is DAG
- $|V| = \Theta(n)$ $|E| = \Theta(n \cdot k)$

$|V| + |E| = \Theta(n \cdot k)$

S.P. soln. is $\Theta(n \cdot k)$

⑤ Pretty Printing Problem

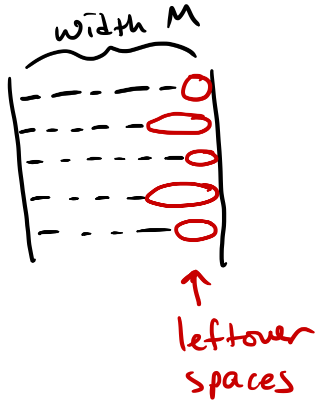
- words of length $l_1\ l_2 \dots l_n$
- line width $= M$
- penalty for placing words $i \to j$ on a line
  = cube # leftover spaces

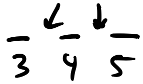
$$= \left(M - \left[\sum_{k=i}^{j} l_k + (j-i)\right]\right)^3$$

$|V| = \Theta(n)$

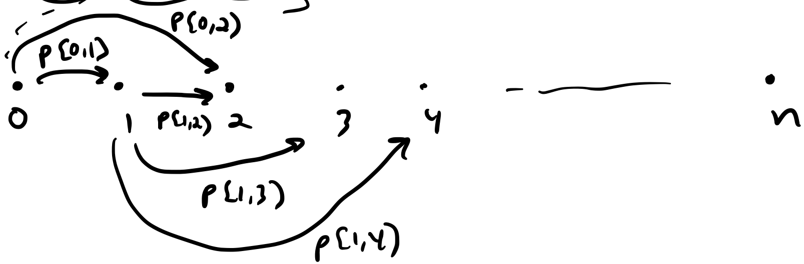
$|E| = O(n \cdot M)$

$|V| + |E| = \Theta(n \cdot M)$





penalty = sum of the cubes of leftover spaces.



vertices = words

edge from $i \to j$: put words $i+1$ to j on a line

edge weight : penalty for that choice

$P[i,j) =$ penalty of words $i+1$ to j on line

- every soln. to P.P.P. is a path in graph
- cost of soln. is path length
- min cost soln. is S.P.

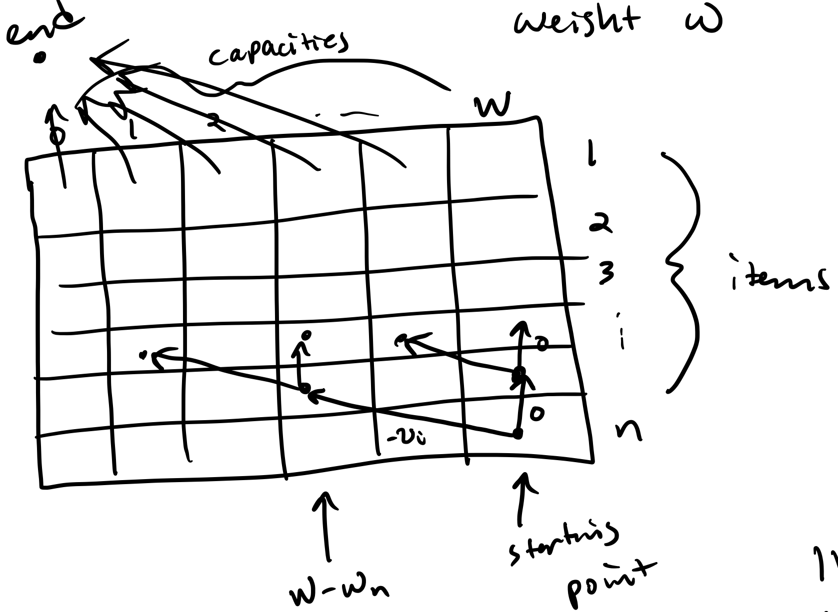
(6) 0-1 Knapsack

items $1, 2, \ldots, n$

w_i = weight of i
v_i = value of i

subproblem for opt. substructure

$[i, w]$ – Knapsack problem w/ items $1 \to i$ and Knapsack weight w



- every soln to Knapsack is a path in graph
- Value of soln. is the negative of the path length
- shortest path is opt. soln.

$|V| = \Theta(nW)$

$|E| = \Theta(nW)$

$|V| + |E| = \Theta(nW)$, s.p. running time

Summarizing

(1) Vertices in graph correspond to objects or

cities in rental car

squares in chessboard problem

Subproblems to problem

[i,w] in 0-1 Knapsack

remaining change in coin changing

(2) edges correspond to choices that can be made

Graph		Problem
path	⟺	valid solution

(3) edge weights correspond to cost (or negative value) of choice

Graph		Problem
weight of path	⟺	cost of solution

(4) shortest path ⟺ opt. solution

(5) If graph is a DAG, running time is linear in size of graph