ALGORITHMIC PARADIGMS

DIVIDE & CONQUER:

Can I split the problem into independent pieces, solve them recursively, and combine answers?

Example: Merge Sort

GREEDY: If I make the best local choice now, am I guaranteed a global optimum?

Example: Dykstra's algorithm

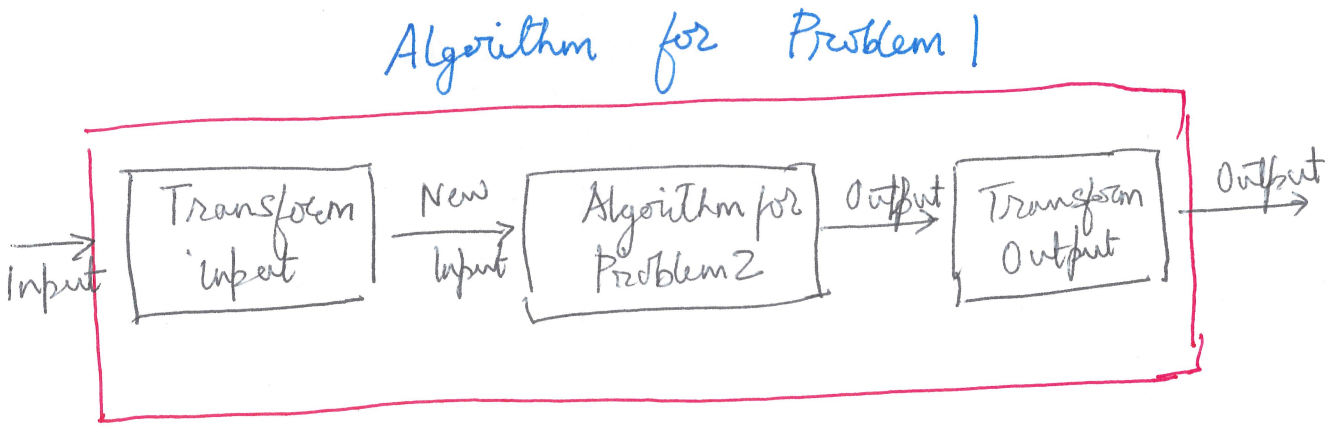
DYNAMIC PROGRAMMING:

Does the problem have overlapping subproblems and the optimal solution built from optimal subsolutions?

Example: Longest Increasing Subsequence

REDUCTIONS

Can I turn this into a problem I already know how to solve?



MEDIAN

FIND MINIMUM: Given a list L of numbers find the smallest number in L.

middle number

$\downarrow L$

SORTING: Given a list L of numbers, rearrange them in ascending order.

$\downarrow L'$

return $L'[0]$ / return $L'[mid]$

HAMILTONIAN PATH IN DAGs:

Given a DAG G does G have a path that visits every vertex

↓ G

UNIQUE TOPOLOGICAL SORT: Given a DAG G, does G have a unique topological sort for the vertices.

↓

return same answer.

If G has a path that visits every vertex

$v_1 \to v_2 \to v_3 \to \ldots v_n$

$\Rightarrow$ the unique ordering is $v_1, v_2 \ldots v_n$

Suppose $v_1, v_2, v_3 \ldots v_n$ is a topological ordering.
If G has a unique ordering then $\forall i$ $v_i \to v_{i+1}$ must be an edge.

REACHING TARGET: Given a directed graph $G = (V, E)$, determine if there is path from every vertex to a vertex $t \in V$.

↓ Reverse G, t

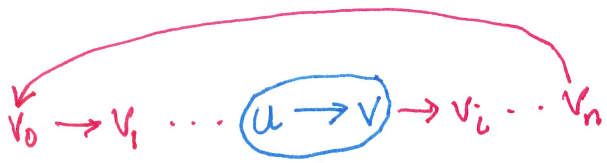
REACHING FROM SOURCE: Given a directed graph $G = (V, E)$ and vertex s, determine if there is path from s to every vertex in G.

Algo: DFS/BFS ~~and~~ from s and check if all vertices are visited

↓

return same answer.

SHORTEST CYCLE: Given a directed graph $G = (V, E)$ and edge $e = (u, v) \in E$, determine the length of shortest cycle that contains $e = (u, v)$.

Cycle containing (u, v): 

path v to u + edge (u, v)

$\downarrow$ $G - (u, v)$, v

SHORTEST PATH: Given a directed graph $G = (V, E)$ and vertex s, compute the length of shortest path from s to every other vertex.

Algo: Run BFS from s.

$\downarrow$ dist

return (dist [u] + 1)

SHORTEST s-t PATH WITHOUT EDGE:

Given directed graph $G = (V, E)$, vertices s and t, and edge $e = (u, v) \in E$, compute the length of the shortest s-t path that does not involve edge $e = (u, v)$.

$\downarrow$ $G - (u, v), s$

SHORTEST s-t PATH: Given directed graph G, vertices s and t, compute the length of shortest path from s to t.

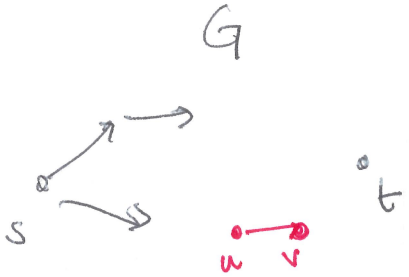
Algo: BFS from s and return distance to t.

$\downarrow$

return same output.

SHORTEST s-t PATH THAT USES EDGE:

Given directed graph $G = (V, E)$, vertices s and t, and edge $e = (u, v) \in E$, what is the shortest path from s to t that uses edge $e = (u, v)$?



Find shortest path from s to u using BFS

Find shortest path from v to t using BFS.

G, s ↓ ↓ G, v

SHORTEST PATH:

dist1 ↓ ↓ dist2

Return dist1[u] + 1 + dist2[t]

ELEMENT DISTINCTNESS:

Given a list L of numbers, are all the numbers in L distinct?

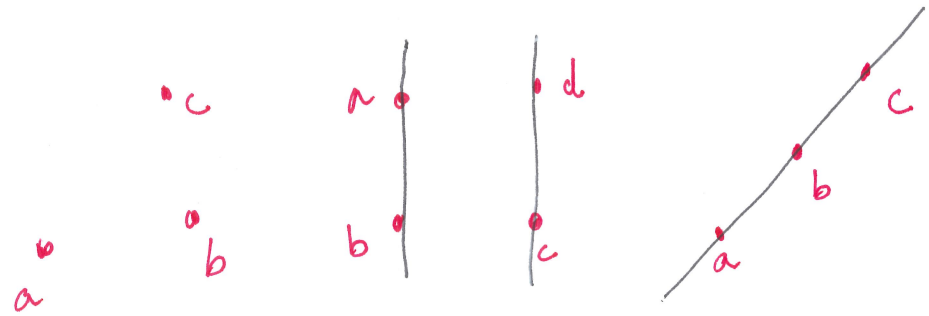
$\downarrow L$

SORTING: Given a list L of numbers, rearrange the numbers in ascending order.

$\downarrow L'$

Check if L' has successive elements that are the same.

COLINEAR : Three points a, b, c on a plane are colinear if you can draw a straight line through all 3 points.



a, b, c are colinear if slope (line a - b) = slope (line b - c)

$a = (x_a, y_a)$, $b = (x_b, y_b)$ $c = (x_c, y_c)$

$\text{slope}(a-b) = \frac{y_a - y_b}{x_a - x_b}$, $\text{slope}(b-c) = \frac{y_b - y_c}{x_b - x_c}$.

L is a list of pairs of numbers

COLINEARITY: Given a list of points L in the 2-dimensional plane and a point p, (→ pair of numbers) determine if there are 2 other points in L that are colinear with p.

$$L' = \{ \text{slope}(p - a) \mid a \in L \}.$$

↓

ELEMENT DISTINCTNESS: