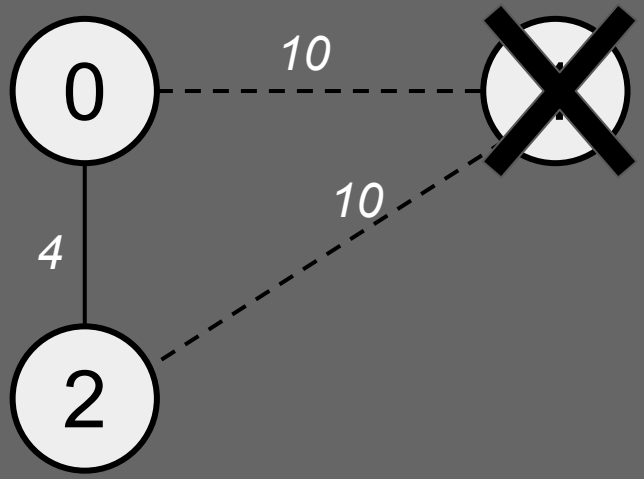


LeetCode Contest Dec 9 Question 4

Problem Definition

Given:

- Set of shops (“branches”)
- Length of roads between these shops
- An integer maxDistance

Find the set of all possible sets of shops you can close to make it such that ***no two shops are more than maxDistance apart***.

Brute Force

1. Generate all possible ***combinations*** of nodes
2. Check each combination to see if it's valid via the shortest path algorithm

- Power set
- Floyd-Warshall
- Slow??? (to run) -> Big-O complexity O(2^n * n^3)
- *(but VERY fast to write!)*

Dynamic Programming

1. Generate combinations as a ***tree***.
2. Each branch adds a new node – update matrix ***incrementally***

- Uses an inefficiency in the brute force method
- Fast to run (100x faster!)
- Slow to write

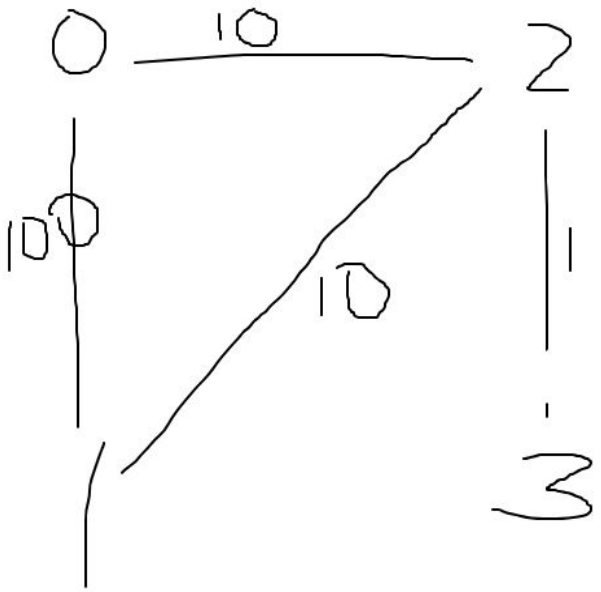
Soln 1: Brute Force (Pseudo-code)

```
all_combos = powerset(n)  # {(),(0),(1)...(1,2),(2,3)

for group in all_combos:
  # calculate the shortest paths from all vertices
  #   - Floyd-Warshall algorithm

  # check if any two vertices are > maxDistance
  for a in group:
    for b in group:
      if distance(a, b) > maxDistance:
        valid = False
  if valid:
    # Add group to the set of 'good' groups
```

Soln 1: Brute Force (Adjacency matrix)

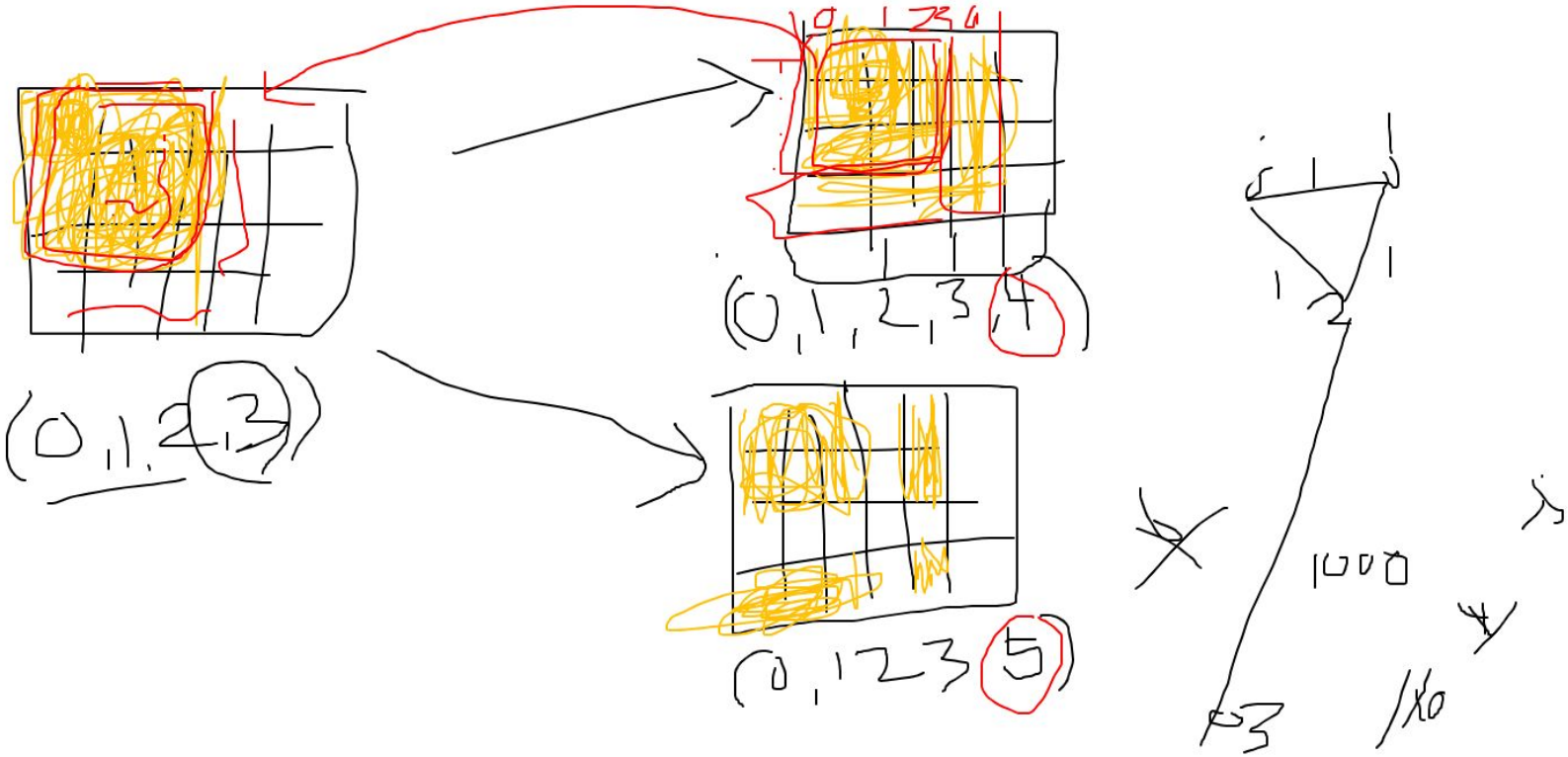


	0	1	2	3
0	0	100	10	x
1	100	0	10	11
2	10	10	0	1
3	x	11	1	0

Soln 2: Dynamic Programming (Pseudo-code)

1. Start from {}
2. Pick a node n that makes a valid graph when added
3. Set is now {n}
4. Update the matrix to include n's neighbors
5. Explore:
    a. All groups with n
    b. All groups without n

Soln 2: Dynamic Programming



Soln 2: Dynamic Programming

- Exploits two inefficiencies with the brute force method:
  - Calculates shortest path faster. Instead of starting from scratch, we update the previous adjacency matrix with just one more node
  - Invalid groups are dropped early. For instance, if {1,2,3} is invalid, and there is no new node that makes it valid, we never explore the rest of the sub-tree

Assumes that adding one node at a time will either result in a valid group, else any combination containing the group is invalid!