Video 42 (1)

Conceptos Principales

1. a
2. b
3. c

Notas

Resumen del algoritmo de Floyd (caminos más cortos "todos contra todos")

La idea de Floyd es encontrar **las distancias más cortas entre *todos* los pares de nodos** de un grafo valorado. A diferencia de Dijkstra, que va de un nodo origen hacia todos los demás, aquí se calcula todo en un solo proceso.

Lo importante es que **se trabaja directamente sobre la matriz de adyacencia**, no sobre listas de nodos ni tablas grandes. Eso lo hace muy eficiente cuando querés saber todas las distancias posibles entre todos los nodos.

Cómo se representa el grafo

Se usa una *matriz de pesos*:

- Si dos nodos están conectados, ponés el peso de esa arista.
- Si no están conectados, ponés un infinito (o algo que represente “no hay camino”).
- Si el grafo no es dirigido, la matriz queda simétrica (peso[i][j] = peso[j][i]).

Qué hace el algoritmo exactamente

La lógica de Floyd es súper directa:

Para cada par de nodos A y B, pregunta:

¿Es más corto ir directo de A a B, o es más corto pasar por algún nodo intermedio K?

Entonces para cada A, B y K hace esta operación:

```
dist[A][B] = min( dist[A][B], dist[A][K] + dist[K][B] )
```

Esa es toda la magia.

Lo que va haciendo es revisar si aparece un camino más barato cuando se usa un nodo intermedio. Si sí lo hay, lo actualiza.

Por qué esto sirve

Porque al terminar, la matriz queda con:

- La distancia mínima entre *todos* los pares.
- Todo calculado en un solo proceso.
- Sin necesidad de ejecutar Dijkstra varias veces.

Esto es útil cuando un programa necesita:

- Saber todas las rutas posibles de forma rápida.
- Resolver problemas tipo “qué tan lejos están todos los nodos entre sí”.
- Evaluar costos globales, no solo desde un punto.

Tabla extra para caminos

Además de la matriz de distancias, Floyd usa una tabla adicional para saber *por dónde se pasa* cuando se toma un camino más barato.

Esta tabla guarda, para cada par de nodos, cuál fue el nodo intermedio que permitió bajar el costo.

Luego reconstruir el camino completo es fácil:

Seguís la cadena de nodos intermedios desde el origen hasta el destino.

Detalles importantes para examen o tareas

- Floyd funciona muy bien en grafos densos o cuando querés muchas consultas de distancia entre pares.
- Su complejidad es $O(n^3)$, pero como trabaja sobre matrices es muy directo de implementar.
- Se puede hacer con una sola matriz si se ordenan bien los ciclos (A, B, K).
- Infinito debe ser un valor especial que nunca podría ser un peso real.
- El algoritmo compara caminos directos contra caminos indirectos usando un nodo intermedio.

En pocas palabras

Floyd revisa todas las combinaciones de caminos posibles usando todos los nodos como intermediarios, y se queda con la opción más barata para cada par de nodos. Al final te deja una matriz con todos los caminos más cortos y una tabla para reconstruirlos.