

# Reporte Técnico - Floyd–Warshall

## Resumen técnico

- Problema: calcular todas las parejas de caminos mínimos (all-pairs shortest paths) en un grafo dirigido/inducido mediante la matriz de adyacencia.
- Supuestos: entrada en forma de matriz de adyacencia  $n \times n$ . Se usa INF para representar ausencia de arista. Los nodos son  $1 \dots n$ . Pesos permitidos: positivos y cero.
- Entregable: programa en C que calcula la matriz de distancias mínimas D y una matriz next para reconstrucción de rutas.

## Desarrollo / Metodología

- Representación: la matriz de distancias D se guarda como `ll **` (matriz dinámica). La matriz `next` es `int **` donde `next[i][j]` es el siguiente vértice tras i en el camino hacia j.
- Inicialización: `D[i][i] = 0`. Si existe arista  $i \rightarrow j$  con peso w, `D[i][j] = w`, si no `D[i][j] = INF`.
- Reconstrucción de rutas: función `reconstruct_path(next, u, v)` que retorna el vector de nodos desde u a v usando `next`.
- Consideraciones INF/desbordamientos: se emplea `INF = 1e15` (constante `long long`) para evitar sumas que desborden en grafos con pesos razonables. Se chequea `dist[i][k] >= INF` o `dist[k][j] >= INF` antes de sumar.

## Verificación de complejidad

- Tiempo: el algoritmo principal realiza 3 bucles anidados sobre  $n$  por lo que su complejidad temporal es  $O(n^3)$ .
- Espacio: se usan dos matrices  $n \times n$ , por lo que la complejidad espacial es  $O(n^2)$ .

## Corridas de ejemplo (instrucciones para generar evidencias)

Compilar:

```
mkdir -p bin  
gcc -o bin/floyd src/*.c -Wall -O2
```

Ejecutar con un caso de prueba de `tests/test1.txt`:

```
./bin/floyd tests/test1.txt
```

El programa imprimirá: - Matriz inicial - Matriz final de distancias D - Matriz next - Hasta 3 rutas reconstruidas

(Incluir capturas de terminal o redirigir salida a un archivo: `./bin/floyd tests/test1.txt > salida_test1.txt` y adjuntar ese archivo como evidencia.)

## Corridas de ejemplo incluidas

Se incluyen tres archivos de prueba en la carpeta `tests/`: - `test1.txt` : grafo de 4 nodos con varias conexiones. - `test2.txt` : grafo con componentes desconectadas. - `test3.txt` : grafo con pesos cero y caminos dirigidos.

Puedes ejecutar cada uno y guardar la salida para las evidencias.

### **Conclusiones**

Floyd-Warshall es un algoritmo sencillo y elegante para obtener distancias mínimas entre todos los pares en grafos pequeños o medianos. Su principal ventaja es la implementación directa usando una matriz de adyacencia y la capacidad de reconstruir rutas usando una matriz `next`. Sin embargo, su complejidad temporal  $O(n^3)$  lo hace inadecuado para grafos muy grandes. En la implementación aprendí a manejar matrices dinámicas en C, reconocer problemas con representación de INF y cómo reconstruir rutas eficientemente.