Algoritmos de Flujo Máximo

Edmonds-Karp vs Ford-Fulkerson

Steven Navarro - 20221020048
Juan Diego Grajales Castillo - 20221020128
Daniel Mateo Ballesteros Molina - 20221020102
Hana Sofia Pinilla Manrique - 20221020092

7 de junio de 2025

Problema a Resolver

Flujo Máximo en Redes

Definición del problema

Dado un grafo dirigido con capacidades en las aristas, encontrar el máximo flujo que puede pasar desde un nodo fuente (source) hasta un nodo sumidero (sink).

Algoritmo Ford-Fulkerson

Enfoque con DFS

Características

- Búsqueda en profundidad (DFS)
- Complejidad: $O(E \cdot f)$
- Flujo entero garantizado
- Fácil implementación

Limitaciones

- Rendimiento depende del tamaño del flujo máximo
- Puede ser lento con capacidades irracionales

Implementación Ford-Fulkerson

```
class FordFulkerson:
    def dfs(self, u, sumidero, flujo_camino):
        self.visitado.add(u)
        if u == sumidero:
            return flujo_camino
        for v in self.grafo.aristas[u]:
            capacidad_residual = self.grafo.capacidad_residual(u, v)
            if v not in self.visitado and capacidad_residual > 0:
                self.padre[v] = u
                flujo = min(flujo_camino, capacidad_residual)
                resultado = self.dfs(v, sumidero, flujo)
                if resultado > 0:
                    return resultado
        return 0
```

Algoritmo Edmonds-Karp

Enfoque con BFS

Características

- Búsqueda en anchura (BFS)
- Complejidad: $O(V \cdot E^2)$
- Camino más corto en cada iteración
- Rendimiento garantizado

Ventajas

- Independiente del tamaño del flujo
- Siempre encuentra solución óptima
- Mejor para grafos complejos

Implementación Edmonds-Karp

```
class EdmondsKarp:
    def bfs(self, fuente, sumidero):
        visitado = set()
        cola = deque([fuente])
        visitado.add(fuente)
        self.padre = {fuente: None}
        while cola:
            u = cola.popleft()
            for v in self.grafo.aristas.get(u, {}):
                if v not in visitado and
                    self.grafo.obtener_capacidad(u, v) > 0:
                    self.padre[v] = u
                    if v == sumidero:
                        return True
                    visitado.add(v)
                    cola.append(v)
        return False
```

Comparación de Algoritmos

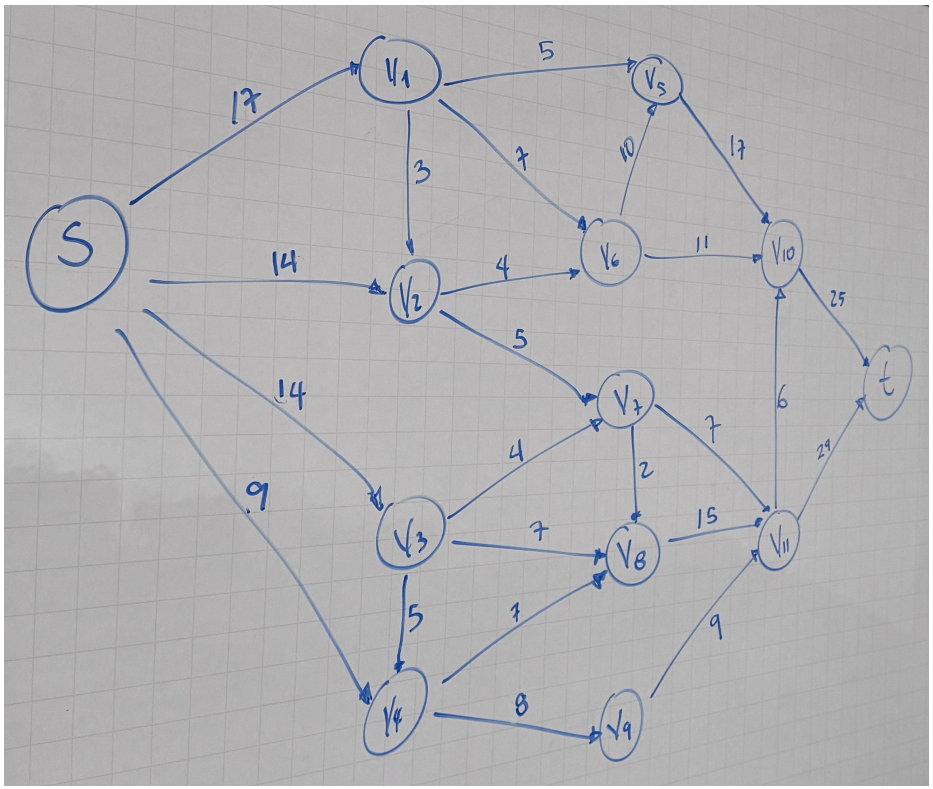
Característica	**Ford-Fulkerson (DFS)**	**Edmonds-Karp (BFS)**
Estrategia	Camino cualquiera	Camino más corto
Complejidad	$O(E \cdot f)$	$O(V \cdot E^2)$
Implementación	Más simple	Más robusta
Flujo máximo		
Aristas reversas	Flujos negativos	Capacidad 0
Rendimiento	Variable	Predecible
Uso memoria	Menor	Mayor

Selección de algoritmo

- **Ford-Fulkerson**: Grafos pequeños con flujos pequeños
- **Edmonds-Karp**: Grafos grandes o con caminos complejos

Ejemplo de Grafo

Red de flujo de entrada



Datos de entrada

- 13 nodos (0-12)
- 24 aristas
- Fuente: nodo 0
- Sumidero: nodo 12

Aristas clave

- $0 \rightarrow 1$ (17)
- $0 \rightarrow 2$ (14)
- $0 \rightarrow 3$ (14)
- $10 \rightarrow 12$ (25)
- $11 \rightarrow 12$ (29)

Resultados de Ejecución

Ford-Fulkerson (DFS)

- Iteración 1: Camino $0 \rightarrow 1 \rightarrow 5 \rightarrow 10 \rightarrow 12$ (5)
- Iteración 2: Camino $0 \rightarrow 3 \rightarrow 8 \rightarrow 11 \rightarrow 12$ (7)
- Iteración 3: Camino $0 \rightarrow 2 \rightarrow 7 \rightarrow 11 \rightarrow 12$ (5)
- Iteración 4: Camino $0 \rightarrow 4 \rightarrow 9 \rightarrow 11 \rightarrow 12$ (8)
- **Flujo máximo: 25**

Edmonds-Karp (BFS)

- Iteración 1: Camino $0 \rightarrow 1 \rightarrow 2 \rightarrow 6 \rightarrow 10 \rightarrow 12$ (4)
- Iteración 2: Camino $0 \rightarrow 1 \rightarrow 6 \rightarrow 10 \rightarrow 12$ (7)
- Iteración 3: Camino $0 \rightarrow 2 \rightarrow 6 \rightarrow 10 \rightarrow 12$ (4)
- Iteración 4: Camino $0 \rightarrow 3 \rightarrow 7 \rightarrow 11 \rightarrow 12$ (4)
- **Flujo máximo: 25**

Conclusión

Ambos algoritmos encuentran el mismo flujo máximo (25), pero recorren caminos diferentes debido a sus estrategias de búsqueda.

Conclusiones

Hallazgos principales

- Ambos algoritmos resuelven eficientemente el problema de flujo máximo
- La implementación con BFS (Edmonds-Karp) ofrece mejor rendimiento garantizado
- La implementación con DFS (Ford-Fulkerson) es más simple de entender
- La elección depende de las características de la red

Mejoras implementadas

- Registro explícito de flujos
- Actualización bidireccional de aristas
- Manejo robusto de archivos de entrada
- Visualización de caminos aumentantes