

Flujos: Network Flow

MSc Edson Ticona Zegarra

Taller avanzado 2025

Contenido

Flujos

Teorema Max-Flow Min-Cut

Flujo Máximo

- ▶ Dado un grafo dirigido con pesos que representan la capacidad de cada *arco*, y un par de vértices s y t , de origen y de destino respectivamente, se busca hallar la mayor cantidad de volumen que puede pasar por el grafo

Flujo Máximo

- ▶ Dado un grafo dirigido con pesos que representan la capacidad de cada *arco*, y un par de vértices s y t , de origen y de destino respectivamente, se busca hallar la mayor cantidad de volumen que puede pasar por el grafo
- ▶ Todas las capacidades son no negativas, $c(u, v) \geq 0$ y si existe una arista (u, v) entonces no existe una arista (v, u)

Flujo Máximo

- ▶ Dado un grafo dirigido con pesos que representan la capacidad de cada *arco*, y un par de vértices s y t , de origen y de destino respectivamente, se busca hallar la mayor cantidad de volumen que puede pasar por el grafo
- ▶ Todas las capacidades son no negativas, $c(u, v) \geq 0$ y si existe una arista (u, v) entonces no existe una arista (v, u)
- ▶ Para todo vértice v , se asume que existe un camino de s a v y de v a t

Flujo Máximo

- ▶ Dado un grafo dirigido con pesos que representan la capacidad de cada *arco*, y un par de vértices s y t , de origen y de destino respectivamente, se busca hallar la mayor cantidad de volumen que puede pasar por el grafo
- ▶ Todas las capacidades son no negativas, $c(u, v) \geq 0$ y si existe una arista (u, v) entonces no existe una arista (v, u)
- ▶ Para todo vértice v , se asume que existe un camino de s a v y de v a t
- ▶ Para todo vértice $v \in V$ excepto s y t , la cantidad de flujo entrante es igual a la cantidad de flujo saliente. Es decir hay una **conservación de flujo**

Ford-Fulkerson

- ▶ Es un método genérico que incrementa iterativamente el flujo, comenzando en 0

Ford-Fulkerson

- ▶ Es un método genérico que incrementa iterativamente el flujo, comenzando en 0
- ▶ En cada iteración incrementa el flujo buscando un *augmenting path* (“camino de aumento”) en una red residual asociada G_f .

Ford-Fulkerson

```
while Exista un augmenting path en la red residual do  
  | Aumentar el flujo  
end
```

Grafo Residual

- ▶ La red residual G_f representa que tanto el flujo puede cambiar en G

Grafo Residual

- ▶ La red residual G_f representa que tanto el flujo puede cambiar en G
- ▶ Dado un arco (u, v) , la *capacidad residual* está dada por $c_f(u, v) = c(u, v) - f(u, v)$

Grafo Residual

- ▶ La red residual G_f representa que tanto el flujo puede cambiar en G
- ▶ Dado un arco (u, v) , la *capacidad residual* está dada por $c_f(u, v) = c(u, v) - f(u, v)$
- ▶ Formalmente:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{si } (u, v) \in E \\ f(v, u) & \text{si } (v, u) \in E \\ 0 & \text{caso contrario.} \end{cases}$$

Augmenting Path

- Un *augmenting path* es un camino desde s hasta t en la red residual G_f

Ford-Fulkerson

```
while Exista un augmenting path en la red residual do  
  |  
   $c_f(p) = \min\{c_f(e) : e \in p\}$   
  for  $(u, v) \in p$  do  
    |  
    if  $(u, v) \in E(G)$  then  
      |  $(u, v).f \leftarrow (u, v).f + c_f(p)$   
    end  
    else  
      |  $(v, u).f \leftarrow (v, u).f - c_f(p)$   
    end  
  end  
end
```

Edmonds-Karp

- ▶ Se calcula el camino mínimo considerando el menor número de aristas y no el peso de estas

Edmonds-Karp

- ▶ Se calcula el camino mínimo considerando el menor número de aristas y no el peso de estas
- ▶ Complejidad: $O(VE^2)$

Edmonds-Karp

```
while Exista un augmenting path en la red residual sin  
    considerar pesos do  
    |   Aumentar el flujo  
end
```

Algoritmo de Dinitz (Dinic)

- ▶ Es similar a Edmonds-Karp pero Dinic es más eficiente.

Algoritmo de Dinitz (Dinic)

- ▶ Es similar a Edmonds-Karp pero Dinic es más eficiente.
- ▶ Se crea un grafo de niveles G_L al recorrer G_f con un BFS

Algoritmo de Dinitz (Dinic)

- ▶ Es similar a Edmonds-Karp pero Dinic es más eficiente.
- ▶ Se crea un grafo de niveles G_L al recorrer G_f con un BFS
- ▶ En el grafo de niveles G_L se busca un *blocking flows* (flujos bloqueante). Un *blocking flow* es aquel que no permite enviar más flujo de s a t

Algoritmo de Dinitz (Dinic)

- ▶ Es similar a Edmonds-Karp pero Dinic es más eficiente.
- ▶ Se crea un grafo de niveles G_L al recorrer G_f con un BFS
- ▶ En el grafo de niveles G_L se busca un *blocking flows* (flujos bloqueante). Un *blocking flow* es aquel que no permite enviar más flujo de s a t
- ▶ Para ello se utiliza un DFS en el grafo de niveles G_L

Algoritmo de Dinitz (Dinic)

- ▶ Es similar a Edmonds-Karp pero Dinic es más eficiente.
- ▶ Se crea un grafo de niveles G_L al recorrer G_f con un BFS
- ▶ En el grafo de niveles G_L se busca un *blocking flows* (flujos bloqueante). Un *blocking flow* es aquel que no permite enviar más flujo de s a t
- ▶ Para ello se utiliza un DFS en el grafo de niveles G_L
- ▶ Complejidad: $O(V^2E)$

Dinitz

```
while Exista un augmenting path en la red residual sin  
  considerar pesos do  
  |  $G_L \leftarrow BFS(G_f)$   
  |  $f' \leftarrow DFS(G_L)$  /* blocking flow */  
  | Aumentar el flujo en  $f'$   
end
```

Contenido

Flujos

Teorema Max-Flow Min-Cut

Dualidad

- ▶ Cuando hablamos de problemas de optimización, el concepto de *dualidad* señala que un problema puede ser entendido de dos maneras: como una maximización o como una minimización

Dualidad

- ▶ Cuando hablamos de problemas de optimización, el concepto de *dualidad* señala que un problema puede ser entendido de dos maneras: como una maximización o como una minimización
- ▶ Si existe dualidad entonces llamamos a uno de ellos como el *problema primal* y al otro como el *problema dual*

Dualidad

- ▶ Cuando hablamos de problemas de optimización, el concepto de *dualidad* señala que un problema puede ser entendido de dos maneras: como una maximización o como una minimización
- ▶ Si existe dualidad entonces llamamos a uno de ellos como el *problema primal* y al otro como el *problema dual*
- ▶ Entonces, si el problema primal es un problema de maximización, el dual será un problema de minimización

Dualidad

- ▶ Cuando hablamos de problemas de optimización, el concepto de *dualidad* señala que un problema puede ser entendido de dos maneras: como una maximización o como una minimización
- ▶ Si existe dualidad entonces llamamos a uno de ellos como el *problema primal* y al otro como el *problema dual*
- ▶ Entonces, si el problema primal es un problema de maximización, el dual será un problema de minimización
- ▶ Si se dice que existe una *dualidad fuerte*, entonces la solución del dual es también la solución del primal

Max-Flow Min-Cut

- El problema del corte mínimo, o *minimum cut* es aquel en el cual se busca una partición tal que la suma de los pesos de las aristas que cortan dicha partición es mínima

Max-Flow Min-Cut

- ▶ El problema del corte mínimo, o *minimum cut* es aquel en el cual se busca una partición tal que la suma de los pesos de las aristas que cortan dicha partición es mínima
- ▶ Formalmente, si V_1 y V_2 son particiones de V , entonces se busca $\min \sum_e w_e$ tal que $e = (u, v); u \in V_1$ y $v \in V_2$

Max-Flow Min-Cut

- ▶ El problema del corte mínimo, o *minimum cut* es aquel en el cual se busca una partición tal que la suma de los pesos de las aristas que cortan dicha partición es mínima
- ▶ Formalmente, si V_1 y V_2 son particiones de V , entonces se busca $\min \sum_e w_e$ tal que $e = (u, v); u \in V_1$ y $v \in V_2$
- ▶ El problema del flujo máximo es el *dual* del problema del corte mínimo, en particular existe una *dualidad fuerte*

Max-Flow Min-Cut

- Para hallar el corte mínimo, al finalizar la ejecución del algoritmo de flujo máximo, se puede recorrer el grafo residual G_f desde s . Todos los vértices que pueden ser alcanzados desde s forman parte de la primera partición, los vértices restantes pertenecen a la otra partición

Max-Flow Min-Cut

- ▶ Para hallar el corte mínimo, al finalizar la ejecución del algoritmo de flujo máximo, se puede recorrer el grafo residual G_f desde s . Todos los vértices que pueden ser alcanzados desde s forman parte de la primera partición, los vértices restantes pertenecen a la otra partición
- ▶ El valor del corte mínimo (suma de pesos de las aristas de van de una partición a otra) es igual al valor del flujo máximo

Contenido

Flujos

Teorema Max-Flow Min-Cut

Emparejamiento

- ▶ Se conoce como *emparejamiento* o *matching* a un conjunto $M \subseteq E$