Flujo

Modelado de Problemas con Max Flow / Min Cut

Lautaro Lasorsa

Universidad de Buenos Aires - FCEN || Universidad Nacional de La Matanza - DIIT

Training Camp 2025

Gracias Sponsors!

Organizador

Diamond Plus

Gold

Outline

Outline

Definición de una red de flujo

Dado un grafo dirigido $G = (V, E)$ con capacidades $c : E \to N$, y dos nodos $S, T \in V$:

- Un flujo $f : E \to N$ es una función que asigna un valor a cada arista, cumpliendo las siguientes condiciones:
  - Capacidad: $0 \leq f(e) \leq c(e)$ para todo $e \in E$.
  - Conservación: Para todo nodo $v \in V - \{S, T\}$ el flujo de entrada y el flujo de salida son iguales:
    - Flujo de entrada: $\sum_{u:(u,v)\in E} f(u, v)$.
    - Flujo de salida: $\sum_{w:(v,w)\in E} f(v, w)$.
- El valor del flujo se define como $|f| = \sum_{(S,v)\in E} f(S, v) = \sum_{(v,T)\in E} f(v, T)$.

Definición del problema de flujo máximo

Dada una red de flujo, consiste en elegir un flujo f que maximice su valor $|f|$, cumpliendo las condiciones de capacidad y conservación.

Theorem

Si todas las capacidades son enteras, entonces existe un flujo máximo f tal que $|f|$ es un entero y $f(e)$ es un entero para todo $e \in E$.

Lo importante, que usaremos para modelado, es que podemos pensar cada unidad de flujo como un cámino que va desde S hasta T, de tal forma que por una arista e no pasan más de $c(e)$ cáminos distintos.
Esto puede verse, por ejemplo, en el problema Distinct Routes (1711) de CSES.

Red residual

Definition (Red residual)

Dado un grafo $G = (V, E)$ con funciones de capacidad c y de flujo f, definimos la red residual $R = (V', E')$ del grafo como:

- Un grafo con el mismo conjunto de nodos V. $V' = V$.
- E' tiene todas las aristas de E con sus mismas capacidades y flujos.
- Para cada arista $e = (u, v) \in E$, existe una arista $e' = (v, u) \in E'$ con $c(e') = 0$ y $f(e') = -f(e)$.

Theorem

Existe un cámino C en R desde S hasta T tal que para cada arista $e \in C$, $f(e) < c(e)$ si y solo si el flujo f no es máximo.
Este cámino C se llama cámino aumentante y enviar una unidad de flujo por C (aumentar en 1 flujo de todas las aristas de C y reducir en 1 el flujo de las aristas inversas) aumenta el valor del flujo $|f|$ en 1.

Si una arista e tiene $f(e) = c(e)$, se dice que está saturada.

Soluciones al problema de flujo máximo

- Ford Fulkerson - Edmonds Karp: En cada iteración busca un cámino aumentante en la red residual utilizando BFS y lo utiliza. Termina cuando no hay más cáminos aumentantes. Es $O(VE^2)$.
- Dinic: Pueden ver los detalles en CP Algorithms. Tiene complejidad general $O(V^2E)$, pero en casos interesantes como redes unitarias o planares es $O(E\sqrt{V})$.
- Programación lineal: Simplex, excesivamente complejo pero interesante desde el punto de vista teórico.

Todos los algoritmos encuentran un flujo máximo f y no solo el valor máximo posible.

Definición de corte

Sea un grafo dirigido ponderado $G = (V, E)$ y dos nodos $S, T \in V$, hay 2 definiciones posibles de corte:

1. Un conjunto de aristas $E' \subseteq E$ tal que si eliminamos todas las aristas de E' del grafo, no hay un cámino desde S hasta T. En este caso, el costo del corte es $c(E') = \sum_{e \in E'} c(e)$.
2. Una partición de los nodos $V = V_1 \cup V_2$ tal que $S \in V_1$ y $T \in V_2$. En este caso, el costo del corte es $c(V_1, V_2) = \sum_{(u,v) \in E, u \in V_1, v \in V_2} c(u, v)$.

Problema de Corte Mínimo

Dado un grafo dirigido ponderado $G = (V, E)$ y un par de nodos $S, T \in V$, el problema de corte mínimo consiste en encontrar un corte de costo mínimo entre S y T.

Teorema de Max Flow - Min Cut

Theorem (Max Flow - Min Cut)

Dado un grafo dirigido ponderado $G = (V, E)$ y un par de nodos $S, T \in V$, el valor del flujo máximo desde S hasta T es igual al costo del corte mínimo entre S y T. (los costos de las aristas se consideran como capacidades)

Theorem

Sea f un flujo máximo en G y R su red residual. Entonces, el conjunto de nodos alcanzables, sin pasar por aristas saturadas, desde S en R forma un lado del corte mínimo, y el conjunto de nodos no alcanzables desde S forma el otro lado del corte mínimo.

Costo de un flujo

Dado un grafo dirigido $G = (V, E)$, dos nodos $S, T \in V$, y las siguientes funciones:

- Capacidad $c : E \to N$.
- Costo $w : E \to R_{\geq 0}$.

Definimos el costo de un flujo f como:

$$C(f) = \sum_{e \in E} w(e) f(e)$$

Problema de Min Cost Max Flow

El problema consiste en encontrar un flujo f de valor máximo, y entre los flujos de valor máximo, el que tenga mínimo costo.
Se resuelve mediante el algoritmo de Edmonds-Karp, pero en lugar de usar BFS, se utiliza un algoritmo de cámino mínimo en grafo ponderado.
Aunque Bellman-Ford es una opción valida, se puede utilizar Dijkstra utilizando una función de potencial para evitar aristas negativas, permitiendo una complejidad de $O(|E||f|\log(|V|))$.

Outline

Definición de Matching

- Sea un grafo $G = (V, E)$, un matching es un conjunto de aristas $M \subseteq E$ tal que no hay dos aristas en M que compartan un nodo. Es decir, cada nodo está emparejado con a lo sumo una arista en M.
- El problema de matching máximo consiste en encontrar un matching M de tamaño (cantidad de aristas) máximo.
- En el caso general se puede resolver con el [algoritmo de Blossom] en $O(|E| * |V|^2)$ y en $O(|E| * |V|^{1/2})$ con el algoritmo de Micali y Vazirani, pero este es demasiado complicado para usarse en programación competitiva.
- Para el caso particular de grafos bipartitos, puede resolver en $O(|E| * \sqrt{|V|})$ utilizando flujo máximo.

Matching Máximo Bipartito como problema de flujo máximo

Sea un grafo bipartito $G = (U \cup V, E)$, donde U y V son los conjuntos de nodos. Definimos el grafo de flujo máximo $G' = (V', E')$ de la siguiente manera:

- Agregamos un nodo fuente S y un nodo sumidero T.
- Para cada nodo $u \in U$, agregamos una arista (S, u) con capacidad 1.
- Para cada nodo $v \in V$, agregamos una arista (v, T) con capacidad 1.
- Para cada arista $(u, v) \in E$, agregamos una arista (u, v) con capacidad ∞.

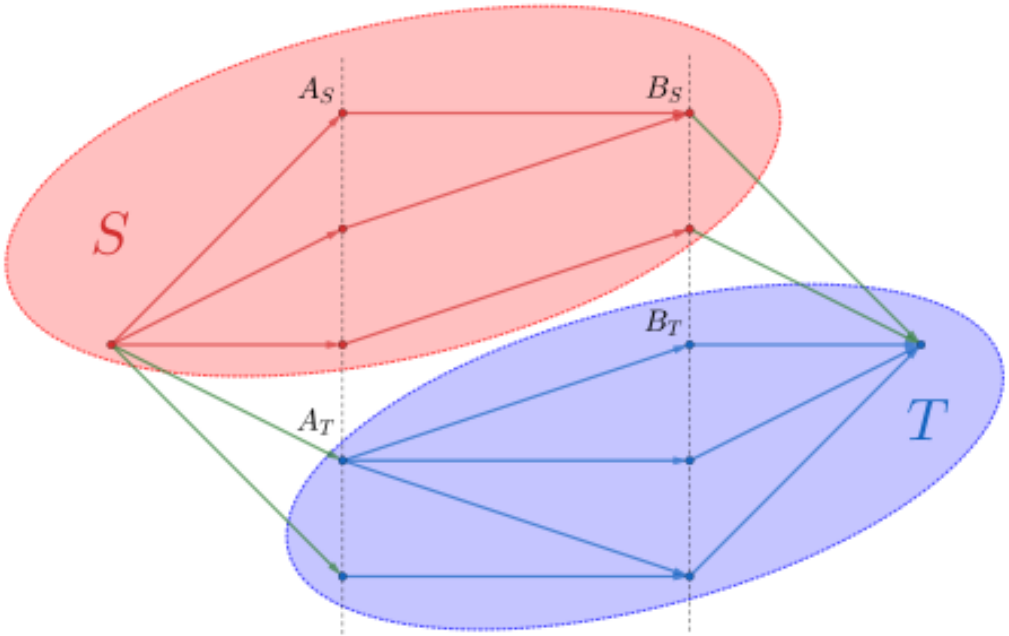
El matching máximo en el grafo original corresponde al flujo máximo en el grafo de flujo máximo, y viceversa.
A su vez, podemos recuperar el matching máximo a partir de la función de flujo f. Para cada arista $(u, v) \in E$, pertenece al matching máximo si y solo si $f(u, v) = 1$.

Propiedades del Matching Máximo Bipartito

Sea $G = (V, E)$, con $V = V_1 \bigcup V_2$, un grafo bipartito y M un matching máximo en G. Entonces:

- Teorema de Konig: $|M|$ es el tamaño del Mínimun Vertex Cover (MVC) de G.
- El MVC está compuesto por los nodos de V_2 alcanzables desde S y los nodos de V_1 que alcanzan a T en la red residual (sin pasar por aristas saturadas).
- En general, $V - MVC = MIS$, donde MIS es el Maximum Independent Set.

Minimum Vertex Cover



Matching Máximo Bipartito Ponderado

Si al grafo bipartito G le agregamos pesos en las aristas, podemos definir el peso de un matching como la suma de los pesos de las aristas utilizadas en él.
Entonces, podemos buscar, entre los matchings de tamaño máximo, aquel que maximice o minimice el peso total.
Esto puede alcanzarse de 2 formas:

1. Agregando el peso de las aristas a la red de flujo y utilizando Min Cost Max Flow en $O(|E| * |f|)$
2. Algoritmo Hungaro: $O(|V|^3)$, con mejor constante que el algoritmo de flujo.

Outline

Partición de un DAG en caminos

Definition

Sea un DAG $G = (V, E)$, podemos buscar una partición de sus nodos en caminos disjuntos. Es decir, buscar caminos $C_1, C-2, ..., C_k \in G$ tales que $V = \bigcup_{i=1}^{k} V(C_i)$ y $C_i \cap C_j = \emptyset$ para todo $i \neq j$.

También podemos buscar una partición no necesariamente disjunta. Esta va a ser equivalente a una partición en caminos disjuntos de la clausura transitiva de G, y es así como la vamos a calcular.
El problema consiste en buscar una partición en caminos de tamaño mínimo, es decir, con la mínima cantidad de caminos.

Reducción a un problema de Matching Máximo

Sea un DAG $G = (V, E)$, para encontrar su mínima partición en caminos disjuntos, podemos reducirlo al problema de Matching Máximo Bipartito:

1. Defino un nuevo grafo bipartito $G' = (V', E')$
2. Para cada nodo $v \in V$, agrego dos nodos $v_{in}, v_{out} \in V'$.
3. Para cada arista $(u, v) \in E$, agrego una arista $(u_{out}, v_{in}) \in E'$.
4. Calculo el Matching Máximo Bipartito M en G'.
5. El tamaño de la mínima partición en caminos será $|V| - |M|$.
6. Las aristas que se usan en los caminos de la partición son las aristas del matching.
7. Para saber donde empiezan los caminos, tomamos los v tales que $v_i n$ no está matcheado.

Outline

Outline