

Optimización en Redes

Dpto. Ingeniería Industrial

Universidad de Chile

20 de marzo de 2012

Contenidos

- 1 Grafos
- 2 Algoritmos específicos para problemas en Grafos

Optimización y Grafos

- Es natural definir problemas sobre redes.
- Muchos problemas se pueden modelar de esta forma.
- Si bien SIMPLEX ofrece un algoritmo para resolverlos, existen mejores algoritmos
 - Mejores en teoría
 - Mejores en la práctica.
 - Menos requerimientos computacionales.
- El uso de estructuras especiales (en este caso las redes) en general permite definir mejores algoritmos prácticos.

Definition (Grafo no dirigido)

Un **Grafo no dirigido** $G = (V, E)$ es un conjunto de **nodos** V y arcos E que son pares no ordenados de nodos.

El arco $u = \{i, j\} \in E$ se dice **incidente** a i (y a j).

Dado $U \subset V$ definimos el **corte** de U como

$$\delta(U) = \{\{i, j\} \in E : i \in U, j \in U^c\}.$$

El **grado** de un nodo v se define como $d(v) = |\delta(\{v\})|$.

Grafos y Definiciones

Definition (Caminos, ciclos y paseos)

Dados $P = \{\{v_j, v_{j+1}\}_{j=1}^{n-1} \subset E$, decimos que P es un **paseo** de v_1 a v_{n+1} .

Si $v_i \neq v_j, \quad \forall i \neq j$, entonces P es un **camino elemental** o **camino**.

Si $v_i \neq v_j, \quad \forall i, j = 1, \dots, n$ y $v_1 = v_{n+1}$, entonces P es un **ciclo elemental** o **ciclo**.

Definition (Conexidad y k-conexidad)

Un Grafo G es conexo si para cualquier $u, v \in V$ existe un camino de u a v en E .

Un Grafo G es k -conexo si para cualquier $\{e_1, \dots, e_{k-1}\} \subset E$ el grafo $G' = (V, E \setminus \{e_1, \dots, e_{k-1}\})$ es conexo.

Definition (Tamaño de un Grafo)

Para un grafo $G = (V, E)$ usualmente denotamos $n = |V|$ y $m = |E|$, y decimos que su **tamaño** es (n, m) .

Grafos y Definiciones

Definition (Grafo dirigido)

Un **Grafo dirigido** $G = (V, E)$ es un conjunto de **nodos** V y arcos E que son pares ordenados de nodos. El arco $u = (i, j) \in E$ se dice **incidente** a i (y a j), i es su **origen** y j es su **destino**. Dado $U \subset V$ definimos el **corte saliente** de U como

$\delta^+(U) = \{(i, j) \in E : i \in U, j \in U^c\}$. Dado $U \subset V$ definimos el **corte entrante** de U como $\delta^-(U) = \{(i, j) \in E : i \in U^c, j \in U\}$. El **grado saliente** (entrante) de un nodo v se define como $d^+(v) = |\delta^+(\{v\})|$ ($d^-(v) = |\delta^-(\{v\})|$).

Definition (Conexidad y k-conexidad)

Un grafo dirigido se dice k -conexo si el grafo no-dirigido correspondiente es k -conexo.

Definition (Flujo)

Dado $G = (V, E)$, $f \in \mathbb{R}_+^E$ y $d \in \mathbb{R}^V$, f es un **flujo** en G satisfaciendo d si $\sum_{e \in \delta^+(v)} f_e - \sum_{e \in \delta^-(v)} f_e = d_v$ para todo $v \in V$.

Grafos y Definiciones

Definition (Caminos, ciclos y paseos)

Dados $P = \{(u_j, v_j) = a_j\}_{j=1}^n \subset E$ y $\{i_j\}_{j=1}^{n+1} \subset V$, decimos que P es un **paseo** de i_1 a i_{n+1} si para todo $j = 1, \dots, n$ se tiene que $a_j = (i_j, i_{j+1})$ (en cuyo caso a_j se dice **directo**) o $a_j = (i_{j+1}, i_j)$ (en cuyo caso a_j se dice **reverso**). Si $i_j \neq i_{j'}$, $\forall j \neq j'$, entonces P es un **camino elemental** o **camino**.

Si $i_j \neq i_{j'}$, $\forall j, j' = 1, \dots, n$ y $i_1 = i_{n+1}$, entonces P es un **ciclo elemental** o **ciclo**.

Definition (Arboles)

Un grafo $G = (V, E)$ es un **árbol** si es conexo y no contiene ciclos. $v \in V$ es una **hoja** si $d(v) \leq 1$.

- Todo árbol posee al menos una hoja.
- G es un árbol si y sólo si $|E| = |V| - 1$ y es conexo.
- Para todo $i, j \in V$ existe un único camino que los une.
- Dado G árbol, si agregamos un arco (distinto), el grafo resultante posee exactamente un ciclo.

Problemas Clásicos en Grafos

Definition (Problema de Distancia Mínima)

Dado $G = (V, E)$ grafo (dirigido), $c \in \mathbb{R}^E$ y $s, t \in V$, ¿Cuál es el camino (dirigido) de distancia mínima de s a t ?

Definition (Problema de flujo máximo)

Dado $G = (V, E)$ grafo dirigido, $u \in \mathbb{R}^E$ y $s, t \in V$, ¿Cuál es el flujo máximo en G satisfaciendo $f_e \leq u_e$?

Definition (Problema de flujo a distancia mínima)

Dado $G = (V, E)$ grafo dirigido, $c, u \in \mathbb{R}^E$ y $d \in \mathbb{R}^V$, ¿Cuál es el flujo $f \leq u$ en G satisfaciendo d de costo mínimo?

- Cada uno de estos problemas podemos escribirlos como LP.
 - ¿Cómo?
- ¿Podremos hacer algo mejor?
- Nos enfocaremos en los dos primeros problemas.

Algoritmos para Camino Mínimo

Theorem (Camino mínimo y Optimalidad)

Sea $G = (V, E)$, $n_o \in V$. Si existen ciclos negativos, no hay solución óptima al problema. Si no, sea p_j^* la distancia mínima de n_o al nodo j , entonces $p_j^* = \min\{p_i + c_{ij} : (i, j) \in \delta^-(j)\}$.

Demostración.

Use dualidad y define $p_{n_o} = 0$.



Algoritmo de Bellman-Ford

Require: $G = (V, E)$, $n_o \in V$, $c \in \mathbb{R}^E$.

- 1: $t \leftarrow 0$, $p_{n_o}^t = 0$, $p_j^t = \infty$
- 2: **repeat**
- 3: $t \leftarrow t + 1$
- 4: **for all** $j \in V \setminus \{n_o\}$ **do**
- 5: $p_j^t = \min\{c_{ij} + p_i^{t-1} : (i, j) \in \delta^-(j)\}$.
- 6: **until** $p^t \neq p^{t-1}$

Algoritmos para Camino Mínimo

Theorem (Correctitud y Complejidad de Bellman-Ford)

Si $G = (V, E)$ con pesos c , el algoritmo de Bellman-Ford termina en no mas de n iteraciones, si no, existe un ciclo negativo.

La **complejidad** del algoritmo es $\mathcal{O}(nm)$.

Algoritmo de Dijkstra

Require: $G = (V, E), o \in V, c \in \mathbb{R}_+^E$

- 1: $N \leftarrow \{o\}, p_i = \infty, \forall v \in V, p_o = 0, S \leftarrow \emptyset$
- 2: **while** $N \neq \emptyset$ **do**
- 3: $j \leftarrow \min\{p_i : i \in N\}$
- 4: $N \leftarrow N \setminus \{j\}, S \leftarrow S \cup \{j\}$
- 5: **for all** $i \in \delta^+(j) \setminus S$ **do**
- 6: $N \leftarrow N \cup \{i\}, p_i = \min\{p_i, p_j + c_{ji}\}$
- 7: **return** $p_i : i \in V$ vector distancias mínimas.

- Algoritmo de Dijkstra es correcto, y puede implementarse en $\mathcal{O}(m \log(n))$

Flujo Máximo

Problema de Flujo Máximo

Dado $G = (V, E)$ grafo dirigido, $u \in \mathbb{R}^E$ y $s, t \in V$, ¿Cuál es el flujo máximo en G satisfaciendo $f_e \leq u_e$?

Definition (Camino Aumentante)

Dado $G = (V, E)$, capacidades $u \in \mathbb{R}^E$, $s, t \in V$ y f flujo factible de s a t . Decimos que un camino P de s a t es **aumentante** si $f_e < u_e$ para los arcos **directos** y $0 < f_e$ para los arcos **reversos**.

$\delta(P) = \min\{u_e - f_e : e \text{ directo}, f_e : e \text{ reverso}\}$ es la **capacidad** de P .

Algoritmo de Ford-Fulkerson

Require: $G = (V, E)$, $u \in \mathbb{R}_+^E$, $s, t \in V$, $f \in \mathbb{R}_+^E$ flujo factible de s a t

- 1: **while** $\exists P$ camino aumentante de s a t **do**
- 2: $f \leftarrow f + \delta(P)u$, donde $u_e = 0$ si $e \notin P$, $u_e = 1$ si e arco directo de P , $u_e = -1$ si e arco reverso de P .
- 3: **return** f flujo máximo

Flujo Máximo

Theorem (Correctitud de Ford Fulkerson)

Si f inicial es entero y $u \in \mathbb{Z}_{+}^E$, el algoritmo de Ford-Fulkerson termina en un número finito de pasos con una solución óptima al problema.

- ¿Cómo buscamos caminos aumentantes?

Algoritmo de Marcado

Require: $s, t \in V, f$ flujo factible

- 1: $I \leftarrow \{s\}, p_s = 0, p_j = \infty \forall j \neq s$
- 2: **while** $I \neq \text{empty}, p_t = \infty$ **do**
- 3: Sea $j \in I, I \leftarrow I \setminus \{j\}$
- 4: **for all** $i \in \delta^+(j), f_{(j,i)} < u_{(j,i)}, p_i = \infty$ **do**
- 5: $a(i) \leftarrow j, p_i = 0, I \leftarrow I \cup \{i\}.$
- 6: **for all** $i \in \delta^-(j), f_{(i,j)} > 0, p_i = \infty$ **do**
- 7: $a(i) \leftarrow j, p_i = 0, I \leftarrow I \cup \{i\}.$

- El algoritmo de marcado toma tiempo $\mathcal{O}(m)$.
- Hay camino aumentante si y solo si $p_t = 0$ (definido por $a(t)$).

Flujo Máximo y Corte Mínimo

Definition (Corte $s - t$)

Un conjunto $S \subset V$ es un **corte $s - t$** si $s \in S, t \notin S$, y su valor es $u(\delta^+(S))$.

Theorem (Max flow-Min Cut)

El valor del flujo máximo de s a t es igual al valor del mínimo corte $s - t$.

- La demostración puede hacerse por el algoritmo (en datos racionales), o por dualidad LP.
- El conjunto de nodos con $p_i = 0$ en la última iteración del algoritmo de marcado es un corte mínimo.
- Si u es entero, la complejidad de Ford-Fulkerson es $\mathcal{O}(mnu_{\max})$.
- Si escogemos siempre el camino aumentante con menos arcos, la complejidad puede llevarse a $\mathcal{O}(m^2n)$.
- Alternativamente, si usamos ideas de **escalamiento**, podemos mejorar la complejidad a $\mathcal{O}(m \log(nu_{\max}))$.