

Resumen de teoremas para el final de Matemática Discreta II

Agustin Curto, agucurto95@gmail.com

2016

Índice general

1. Parte A	2
1.1. La complejidad de EDMONS-KARP	2
1.2. Las distancias de Edmonds-Karp no disminuyen en pasos sucesivos	3
1.3. La complejidad de DINIC	4
1.4. La complejidad de WAVE	5
1.5. La distancia entre NA sucesivos aumenta	5
2. Parte B	6
2.1. 2-COLOR es polinomial	6
2.2. Teorema Max-Flow Min-Cut	6
2.3. Complejidad del Hungaro es $\mathcal{O}(n^4)$	7
2.4. Teorema de Hall	8
2.5. Teorema del matrimonio	9
2.6. Si G es bipartito $\Rightarrow \chi'(G) = \Delta$	9
2.7. Teorema cota de Hamming	9
2.8. Sea H una matriz de chequeo de un código C , pruebe que:	9
2.8.1. $\delta(C)$ = mínimo número de columnas linealmente dependientes de H	9
2.8.2. Si H no tiene la columna cero ni columnas repetidas $\Rightarrow C$ corrige al menos un error	9
2.9. Sea C un código cíclico de dimensión k y longitud n y sea $g(x)$ su polinomio generador, probar que:	9
2.9.1. C está formado por los múltiplos de $g(x)$ de grado menor a n	9
2.9.2. El grado de $g(x)$ es $n - k$	9
2.9.3. $g(x)$ divide a $1 + x^n$	9
3. Parte C	10
3.1. $4\text{-COLOR} \leq_p \text{SAT}$	10
3.2. 3-SAT es NP-Completo	10
3.3. 3-COLOR es NP-Completo	10

Capítulo 1

Parte A

1.1. La complejidad de EDMONS-KARP

Teorema: La complejidad de $\langle E - K \rangle$ con $n = |V|$ y $m = |E|$ es $\mathcal{O}(nm^2)$.

Prueba: Sean: f_0, f_1, f_2, \dots la sucesión de flujos creados por $\langle E - K \rangle$. Es decir, el paso k crea f_k .

Para cada k definimos funciones:

- $d_k(x)$ = “distancia” entre s y x en el paso k en caso de existir, si no ∞ .
- $b_k(x)$ = distancia entre x y t en el paso k en caso de existir, si no ∞ .

“Distancia”: longitud del menor camino aumentante entre dos vértices.

Observaciones:

1.
 - $d_k(s) = 0$
 - $b_k(t) = 0$
2. Sabemos que las distancias de $\langle E - K \rangle$ no disminuyen en pasos sucesivos, como esto será útil en esta demostración llamaremos \otimes a la demostración de:

$$d_k(x) \leq d_{k+1}(x)$$

Llamemos *crítico* a un lado disponible en el paso k pero no disponible en el paso $k+1$. Es decir, si xy es un lado $\Rightarrow xy$ se satura o yx se vacía en el paso k .

Supongamos que al construir f_k el lado xy se vuelve crítico, el camino: $s \cdots x, y \cdots t$ se usa para construir f_k .

$$\begin{aligned} d_k(t) &= d_k(x) + b_k(x) \\ &= d_k(x) + b_k(y) + 1 \end{aligned} \tag{1.1}$$

Para que xy pueda ser *crítico* nuevamente debe ser usado en la otra dirección (*i.e* yx). Sea j el paso posterior a k en el cual se usa el lado en la otra dirección, el camino $s \cdots y, x \cdots t$ se usa para construir f_j .

$$\begin{aligned} d_j(t) &= d_j(x) + b_j(x) \\ &= d_j(y) + 1 + b_j(x) \end{aligned} \tag{1.2}$$

Es decir:

$$\text{De } \textcircled{1} \text{ y } \textcircled{2} \Rightarrow \begin{cases} d_j(x) = d_j(y) + 1 \\ d_k(y) = d_k(x) + 1 \end{cases}$$

Luego:

$$d_j(t) = d_j(x) + b_j(x) \quad (1.3)$$

$$= d_j(y) + 1 + b_j(x) \quad (1.4)$$

$$\geq d_k(y) + 1 + b_k(x) \quad (1.5)$$

$$= d_k(x) + 1 + 1 + b_k(x) \quad (1.6)$$

$$= d_k(t) + 2 \quad (1.7)$$

$$\Rightarrow d_\ell(t) \geq d_k(t) + 2 \quad (1.8)$$

Por lo tanto cuando un lado se vuelve crítico recién puede volver a saturarse cuando la distancia de s a t haya aumentado en por lo menos 2. Puede existir $\mathcal{O}(n/t)$ tales aumentos, es decir:

$$\# \text{ Veces que un lado puede volverse crítico} = \mathcal{O}(n).$$

$$\begin{aligned} \therefore \text{Complejidad}(\langle E - K \rangle) &= (\# \text{pasos}) * \text{Complej}(\text{paso}) \\ &= (\# \text{veces que un lado se vuelve crítico}) * (\# \text{lados}) * \text{Complej}(BFS) \\ &= \mathcal{O}(n) * \mathcal{O}(m) * \mathcal{O}(m) \\ &= \mathcal{O}(nm^2) \end{aligned}$$

1.2. Las distancias de Edmonds-Karp no disminuyen en pasos sucesivos

Teorema: Sean: f_0, f_1, f_2, \dots la sucesión de flujos creados por $\langle E - K \rangle$, i.e el paso k crea f_k . Para cada k definimos funciones:

- $d_k(x)$ = “distancia” entre s y x en el paso k en caso de existir, si no ∞ .
- $b_k(x)$ = distancia entre x y t en el paso k en caso de existir, si no ∞ .

“Distancia”: longitud del menor camino aumentante entre dos vértices.

Queremos probar que:

1. $d_k(x) \leq d_{k+1}(x)$
2. $b_k(x) \leq b_{k+1}(x)$

Prueba: Lo probaremos por inducción y solo para d_k ya que para b_k la prueba es análoga.

$$\text{HI: } H(i) = \{\forall_z : d_{k+1}(z) \leq i, d_k(z) \leq d_{k+1}(z)\}$$

1. $H(0) = \{\forall_z : d_{k+1}(z) \leq 0, d_k(z) \leq d_{k+1}(z)\}$

$$\text{Pero } d_{k+1} \leq 0 \Rightarrow$$

1.3. La complejidad de DINIC

Teorema: La complejidad del algoritmo de Dinic es $\mathcal{O}(n^2m)$.

Prueba: Como la distancia entre s y t en networks auxiliares consecutivos aumenta y puede ir a lo sumo entre 1 y $n - 1$ entonces hay a lo sumo $\mathcal{O}(n)$ networks auxiliares.

Notación: llamemos PB al proceso de hallar paso bloqueante en un network auxiliar con Dinic.

Luego la complejidad de Dinic es $\mathcal{O}(n) \text{ compl(PB)}$. Para probar que la complejidad de Dinic es $\mathcal{O}(n^2m)$ debemos probar que $\text{compl(PB)} = \mathcal{O}(nm)$.

Para hallar un flujo bloqueante:

1. Crear un NA: Como es con BFS es $\mathcal{O}(m)$

2. Hallar bloqueante, denotemos:

- A: avanzar
- R: retorceder
- I: inicializar e incrementar

El paso bloqueante de Dinic luce de la forma:

AA...AIAAARA...AIAARAAARR...IA...

subdiviámoslo en palabras del tipo:

- AA...AI
- AA...AR

donde las primeras son todas A pudiendo ser 0 la cantidad de la misma.

Debemos determinar:

1. Cual es la complejidad de cada subpalabra.

Recordemos que:

- A: $\{ P[i + 1] = \text{algún elemento de } \Gamma^+(P[i]) \}$
 $i = i + 1$
 $\Rightarrow A \text{ es } \mathcal{O}(1)$
- R: $\{ \text{borrar } P[i - 1] \text{ del NA} \}$
 $i = i - 1$
 $\Rightarrow R \text{ es } \mathcal{O}(1)$
- I: $\{ \text{Recorre dos veces un camino de longitud } d = d(t) \}$
 $\Rightarrow I \text{ es } \mathcal{O}(d)$

Por lo tanto:

$$\text{compl}(A...AR) = \mathcal{O}(1) + \dots \mathcal{O}(1) + \mathcal{O}(1) = \mathcal{O}(j) \quad (1.9)$$

Pero como cada A hace $i = i + 1$ y tenemos $0 \leq i \leq d \Rightarrow j \leq d$.

$$\therefore \text{compl}(A...AR) = \mathcal{O}(d)$$

Similarmente:

$$\text{compl}(A...AI) = \mathcal{O}(1) + \dots \mathcal{O}(1) + \mathcal{O}(1) = \mathcal{O}(d) + \mathcal{O}(d) = \mathcal{O}(d) \quad (1.10)$$

Pero como cada A hace $i = i + 1$ y tenemos $0 \leq i \leq d \Rightarrow j \leq d$.

$$\therefore \text{compl}(A...AR) = \mathcal{O}(d)$$

2. Cuantas palabras hay de cada tipo.

1.4. La complejidad de WAVE

1.5. La distancia entre NA sucesivos aumenta

Capítulo 2

Parte B

2.1. 2-COLOR es polinomial

2.2. Teorema Max-Flow Min-Cut

Teorema:

- a) Si f es flujo y S es corte $\Rightarrow V(f) \leq \text{Cap}(S)$.
- b) Si $V(f) = \text{Cap}(S) \Rightarrow f$ es maximal y S es minimal.
- c) Si f es maximal $\Rightarrow \exists S$ con $V(f) = \text{Cap}(S)$.

Prueba: Demostraremos primero que $V(f) = f(S, \bar{S}) - f(\bar{S}, S)$ donde f es un flujo y S un corte. Esto nos ayudará en la demostración del ítem (a).

Observemos que:

- $f(A \cup B, C) = f(A, C) + f(B, C) : A$ y B disjuntos.
- $f(A, B \cup C) = f(A, B) + f(A, C) : B$ y C disjuntos.
- $f(A, B) = \sum_{\substack{x \in A \\ y \in B}} f(x, y)$.

Sea $x \in S \Rightarrow x \neq t$.

$$f(x, V) - f(V, x) = \begin{cases} V(f) & \text{Si } x = s \\ 0 & \text{Si } x \neq s \text{ pues } t \notin S \end{cases}$$

Luego:

$$\sum_{x \in S} (f(x, V) - f(V, x)) = 0 + 0 \cdots + V(f) = V(f) \quad (2.1)$$

$$V(f) = \sum_{x \in S} f(x, V) - \sum_{x \in S} f(V, x) \quad (2.2)$$

$$= f(S, V) - f(V, S) \quad (2.3)$$

$$= f(S, S \cup \bar{S}) - f(S \cup \bar{S}, S) \quad (2.4)$$

$$= f(S, S) + f(S, \bar{S}) - f(S, S) - f(\bar{S}, S) \quad (2.5)$$

a) $V(f) \leq \text{Cap}(S)$

2.3. Complejidad del Húngaro es $\mathcal{O}(n^4)$

Teorema: La complejidad del algoritmo Húngaro es $\mathcal{O}(n^4)$.

Prueba:

1. La complejidad del matching inicial es $\mathcal{O}(n^2)$, ya que:

Restar mínimo de cada fila:

$$(\mathcal{O}(n^2) + \mathcal{O}(n^2)) * n = \mathcal{O}(n^2) \text{ Idem para las columnas.}$$

2. Llamemos **extender** el matching, a incrementar su número de filas en 1, i.e agregar una fila más al matching.

$$\# \text{ extensiones de matching} = \mathcal{O}(n)$$

Resta ver la complejidad de cada **extender**.

3. En cada extensión vamos a ir revisando filas y columnas, donde escanear una fila es $\mathcal{O}(n)$ y se realizan n escaneos, por lo que sería $\mathcal{O}(n^2)$ sin considerar que se debe realizar un cambio de matriz.

Hacer un cambio de matriz es $\mathcal{O}(n^2)$.

- Buscar $m = \min S \times \overline{\Gamma(S)} \rightarrow \mathcal{O}(n^2)$
- Restar m de $S \rightarrow \mathcal{O}(n^2)$
- Sumar m a $\Gamma(S) \rightarrow \mathcal{O}(n^2)$

Luego la implementación NAIVE lanzaría nuevamente el algoritmo desde cero. La forma correcta es continuar con el matching que teníamos, ya que el mismo no se pierde.

$$\begin{bmatrix} A & A \\ B & C \end{bmatrix}$$

TO DO

Debemos ver cuantos Cambios de matriz hay antes de extender nuevamente un matching

Lema Interno: Luego de un cambio de matriz, se extiende el matching (i.e se termina el **extender**), o bien se aumenta S .

Prueba:

$$\begin{bmatrix} A & A \\ B & C \end{bmatrix}$$

Al restar $m = \min S \times \Gamma(S)$ de las filas de S , habrá un nuevo cero en alguna fila $i \in S$ y columna $j \in \Gamma(S)$ entonces la columna se etiquetará con i y se revisará. Tenemos dos resultados posibles:

- a) j está libre (i.e no forma parte del matching) \Rightarrow extendemos el matching.

- b) j forma parte de matching $\Rightarrow \exists$ fila k matcheada con j . En este caso, la fila k se etiquetará con j , por lo que el "nuevo" $S \geq S \cup \{k\}$.

Entonces se termina con una extensión o se produce un nuevo S de cardinalidad, al menos $|S| + 1$.

Fin lema interno

Luego como $|S|$ solo puede crecer $\mathcal{O}(n)$ veces, tenemos que hay a lo sumo n **cambios de matriz** antes de extender el matching. Entonces:

$$\begin{aligned} \text{Complejidad(1 Extensión)} &= \underbrace{\mathcal{O}(n)}_{\#CM} * \underbrace{\mathcal{O}(n^2)}_{\text{Compl}(CM)} + \underbrace{\mathcal{O}(n^2)}_{\text{Busqueda } n \text{ filas } x \text{ } n \text{ columnas}} \\ \therefore \text{Complejidad(Húngaro)} &= \underbrace{\mathcal{O}(n^2)}_{\text{Matching inicial}} + \underbrace{\mathcal{O}(n)}_{\#extensiones} * \underbrace{\mathcal{O}(n^3)}_{\text{Compl}(extension)} \end{aligned}$$

2.4. Teorema de Hall

Teorema: Sea $G = (x \cup y, E)$ grafo bipartito $\Rightarrow \exists$ matching completo de X a $Y \Leftrightarrow |S| = |\Gamma(S)| \forall S \subseteq X$.

Prueba:

\Rightarrow) Si M es matching comple de X a Y entonces observemos que M induce una función inyectiva de X a Y .

$$f(x) = \text{único } y : xy \in M.$$

1. Si $S \subseteq X \Rightarrow |S| = |\Gamma(S)|$.

Además por definición de f , $f(x) \in \Gamma(x)$.

2. Si $x \in S \Rightarrow f(x) \in \Gamma(S) \Rightarrow f(S) \subseteq \Gamma(S)$.

De ① y ② $\Rightarrow |S| \leq |\Gamma(S)|$.

\Leftarrow) Supongamos que no es cierto, entonces G es bipartito con $|S| \leq |\Gamma| \forall S \subseteq X$ pero no tiene matching completo de X a Y . Es equivalente a ver que: Si \nexists un matching completo $\Rightarrow \exists S \subseteq X : |S| > |\Gamma(S)|$.

Corramos el algoritmo para hallar matching. Al finalizar habrá filas sin matcher (las de s). Sean:

■

2.5. Teorema del matrimonio

2.6. Si G es bipartito $\Rightarrow \chi'(G) = \Delta$

2.7. Teorema cota de Hamming

2.8. Sea H una matriz de chequeo de un código C , pruebe que:

2.8.1. $\delta(C) = \text{mínimo número de columnas linealmente dependientes de } H$

2.8.2. Si H no tiene la columna cero ni columnas repetidas $\Rightarrow C$ corrige al menos un error

2.9. Sea C un código cíclico de dimensión k y longitud n y sea $g(x)$ su polinomio generador, probar que:

2.9.1. C está formado por los múltiplos de $g(x)$ de grado menor a n

2.9.2. El grado de $g(x)$ es $n - k$

2.9.3. $g(x)$ divide a $1 + x^n$

Capítulo 3

Parte C

3.1. 4-COLOR \leq_p SAT

3.2. 3-SAT es NP-Completo

3.3. 3-COLOR es NP-Completo

Bibliografía

- [1] CURTO AGUSTÍN , «Matemática Discreta II, apuntes de clase», *FaMAF, UNC*.
- [2] MAXIMILIANO ILLBELE, «Resumen de Discreta II, 16 de agosto de 2012», *FaMAF, UNC*.