# Final Matemática Discreta II
# TEÓRICO
# GRUPO B

Agustín M. Domínguez

Julio 2024

# Índice

# Contexto

## 1.1 Introducción

Este apunte fue construido para final de la materia **Matemática Discreta II** de la **Facultad de Matemática Física y Computación** en las fechas de Julio/Agosto del año 2024.

**Una nota para otros estudiantes:** El objetivo de las demostraciones de **este apunte** es ayudar a entenderlas, no ayudar a memorizarlas. Están escritas con todos los pasos explícitos y estrictos, a veces al punto de redundancia y obviedad, *a propósito*. La idea es que el lector tenga que hacer la menor cantidad de saltos de lógica posibles para entender la demostración. La resolución que se haga en el final probablemente pueda tener algunos pasos menos explícitos, pero no es el marco de este apunte.

Fue declarado por la cátedra que estas preguntas van a formar el Teórico del final. A continuación se repite dicho enunciado.

## 1.2 Enunciado

La parte teórica del final consistirá de 3 preguntas tomadas de esta lista, mas una pregunta extra del tema de Milagro.

Para aprobar el teórico hay que obtener 40% del puntaje EN CADA pregunta (de esta lista). Estos teoremas son para Julio/Agosto 2024, excepto algunos que estan marcado que solo se tomaran a partir de Diciembre.

Ademas de estos teoremas, en Diciembre/Febrero/Marzo pueden agregarse incluso otros, pero si lo hacemos se indicara en la pagina de la materia. Si no se dice nada, estos son los que valen.

De los **tres** teoremas que se tomarán de esta lista, **uno será de los seis primeros, otro de los tres ultimos y el otro del resto**.

## 1.3 Grupo

Este apunte tiene las respuestas de las opciones para la **segunda pregunta**, desde ahora llamadas el **Grupo B**.

# Pregunta 7: Max-Flow Min-Cut

## 2.1 Enunciado detallado

Probar que el valor de todo flujo es menor o igual que la capacidad de todo corte y que si $f$ es flujo, entonces $f$ es maximal si y solo si existe un corte $S$ tal que $v(f) = \text{CAP}(S)$. (y en este caso, $S$ es minimal)

(puede usar sin necesidad de probarlo que si $f$ es flujo y $S$ es corte, entonces $v(f) = f(S, \overline{S}) - f(\overline{S}, S)$)

## 2.2 Conocimiento presupuesto para la resolución

**Notación**: Dada una funcion $g$ definida en $E$, y dados $A, B \subseteq V$, definimos:

$$g(A, B) = \sum_{\substack{x \in A \\ y \in B \\ (x,y) \in E}} g(x, y)$$

Es decir aplicar la funcion $g$ a dos subconjuntos de V es la suma de todos los lados de la network que empiecen en A y terminen n B

*Def:* **Flujo**

Dado una network $N = (V, E, C)$ y vertices $s$ y $t$, un Flujo $s$ a $t$ es una función de los lados con:

1. $0 \leq f(\overrightarrow{xy}) \leq C(\overrightarrow{xy}) \quad \forall \overrightarrow{xy} \in E$
2. $\text{IN}_f(x) = \text{OUT}_f(x) \quad \forall x \neq s, t$
3. $\text{IN}_f(s) = 0 = \text{OUT}_f(t)$

*Def:* **Valor de un Flujo**

$v(f) = \text{OUT}_f(s) - \text{IN}_f(s)$

Un flujo es **Maximal** si $v(g) \leq v(f)$ $\forall g$ flujo de $s$ a $t$

*Def:* **Corte**

Un corte es un $S \subseteq V$ tal que $s \in S, t \notin S$

Por ejemplo $S_1 = \{s\} \quad S_2 = V - \{t\}$ son cortes.

*Def:* **Capacidad de un Corte**

$$\text{CAP}(S) = C(S, \overline{S}) = \sum_{\substack{x \in S \\ y \notin y \\ \overrightarrow{xy} \in E}} C(\overrightarrow{xy})$$

Un corte se dice *minimal* si $\text{CAP}(S) \leq \text{CAP}(T)\ \forall\ T$ corte

*Def:* **Camino Aumentante**

Dado un flujo $f$ en un network, un camino aumentante o $f$-camino aumentante es una sucesión de vértices $x_0, x_1, \ldots, x_r$ con $x_0 = s,\ x_r = t$ donde $\forall i < r$ ocurre una y solo una de las siguientes propiedades:

- $\overrightarrow{x_i x_{i+1}} \in E \ \wedge\ f(\overrightarrow{x_i x_{i+1}}) < C(\overrightarrow{x_i x_{i+1}})$ llamados lados *forward*
- $\overrightarrow{x_{i+1} x_i} \in E \ \wedge\ f(\overrightarrow{x_{i+1} x_i}) > 0$ llamados lados *backward*

## 2.3 Resolución

Esta prueba tiene múltiples propiedades a probar. Separemos y rearmemos las hipótesis y las pruebas:

**(1)** = Dado $f$ flujo y $S$ corte, entonces: $v(f) \leq \text{CAP}(S)$

**(2)** = $f$ flujo es máximal $\iff \exists\ S$ corte $v(f) = \text{CAP}(S)$

**(3)** = ($f$ flujo máximal) $\wedge\ v(f) = \text{CAP}(S) \implies S$ minimal.

Propiedad asumida sin demonstración: Dado $f$ flujo y $S$ corte, entonces:

$$v(f) = f(S, \overline{S}) - f(\overline{S}, S) \quad \textbf{(P0)}$$

*Dem.* Demonstremos **(1)**:

Por definición de flujo tenemos: $0 \leq f(\overrightarrow{xy}) \leq C(\overrightarrow{xy}) \quad \forall \overrightarrow{xy} \in E$

Si aplicamos esa propiedad con la notación $\implies 0 \leq f(\overline{S}, S) \quad \forall S$ corte **(P1)**

Por la propiedad **(P0)**, tenemos:

$$v(f) = f(S, \overline{S}) - f(\overline{S}, S) \overset{\textbf{(P1)}}{\implies} v(f) \leq f(S, \overline{S})$$

Nuevamente por definición de flujo: $f(\overrightarrow{xy}) \leq C(\overrightarrow{xy}) \quad \forall \overrightarrow{xy} \in E$

$\implies f(S, \overline{S}) \leq C(S, \overline{S}) \quad \forall S$ corte.

$\implies \quad v(f) \leq f(S, \overline{S}) \leq C(S, \overline{S}) \overset{\text{Def CAP}}{=} \text{CAP}(S)$

$\therefore \quad v(f) \leq \text{CAP}(S)$ **(1)**

□

*Dem.* Ahora veamos **(2)**:

**Ida** ( $\implies$ ):

Supongamos $f$ flujo es máximal $\implies v(g) \leq v(f) \quad \forall\ g$ flujo

Primero veremos que no puede existir f-camino aumentante.

Si existiese un f-camino aumentante, podríamos mandar un $\varepsilon > 0$ a través de él, y obtener un flujo $f'$ tal que $v(f') = v(f) + \varepsilon$

$\implies v(f') > v(f) \implies$ f no es máximal. **Absurdo**.

$\therefore \quad$ No existe f-camino aumentante. **(P2)**

Sea S = $\{ s \} \cup \{ x \in V \mid$ Exista un f-Camino aumentante desde $s$ a $x$ $\}$

Por **(P2)**, no existe f-camino aumentante $\implies t \notin S \overset{s \in S}{\implies} S$ corte.

$S$ corte $\overset{\textbf{(P0)}}{\implies} v(f) = f(S, \overline{S}) - f(\overline{S}, S)$

Por notación:

$$f(S, \overline{S}) = \sum_{\substack{x \in S \\ y \notin S \\ \overrightarrow{xy} \in E}} f(\overrightarrow{xy})$$

Sea $x, y \mid x \in S \wedge y \notin S \wedge \overrightarrow{xy} \in E$

Por propiedad de flujo: $f(\overrightarrow{xy}) \leq c(\overrightarrow{xy})$.

**Supongamos** $f(\overrightarrow{xy}) < c(\overrightarrow{xy})$

Como $x \in S$, entonces existe un f-camino aumentante entre $s$ y $x$.

Sea ese camino: $s = x_0, x_1, x_2, \ldots, x_r = x$

Luego como $f(\overrightarrow{xy}) < c(\overrightarrow{xy})$ podemos usar $\overrightarrow{xy}$ como lado forward y crear el f-camino aumentante:

$s = x_0, x_1, x_2, \ldots, x_r = x, y \implies$ Existe f-camino aumentante entre $s$ y $y \implies y \in S$

Pero $y \notin S$: **Absurdo.**

$\therefore \quad f(\overrightarrow{xy}) = c(\overrightarrow{xy})$

Esto aplica para todos los términos de $f(S, \overline{S})$. Luego:

$$f(S, \overline{S}) = \sum_{\substack{x \in S \\ y \notin S \\ \overrightarrow{xy} \in E}} f(\overrightarrow{xy}) = \sum_{\substack{x \in S \\ y \notin S \\ \overrightarrow{xy} \in E}} c(\overrightarrow{xy}) = c(S, \overrightarrow{S}) = \text{CAP}(S)$$

$\therefore \quad f(S, \overline{S}) = \text{CAP}(S)$

Veamos ahora $f(\overline{S}, S)$:

$$f(\overline{S}, S) = \sum_{\substack{x \notin S \\ y \in S \\ \overrightarrow{xy} \in E}} f(\overrightarrow{xy})$$

Sea $x, y \mid x \notin S \wedge y \in S \wedge \overrightarrow{xy} \in E$

Por propiedad de flujo: $f(\overrightarrow{xy}) \geq 0$

**Supongamos** $f(\overrightarrow{xy}) > 0$

$y \in S \implies$ Existe f-camino aumentante entre $s$ y $y$

Sea ese camino: $s = y_0, y_1, \ldots, y_j = y$

Luego como $f(\overrightarrow{xy}) \geq 0$, se puede usar el lado $\overrightarrow{xy}$ Backwards, y así obtener el f-camino aumentante:

$s = y_0, y_1, \ldots, y_j = y, x \implies$ Existe f-camino aumentante entre $s$ y $x \implies x \in S$

Pero $x \notin S$. **Absurdo**.

$\therefore \quad f(\overrightarrow{xy}) = 0 \quad \forall\ x, y \mid x \notin S \wedge y \in S \wedge \overrightarrow{xy} \in E$

Luego:

$$f(\overline{S}, S) = \sum_{\substack{x \notin S \\ y \in S \\ \overrightarrow{xy} \in E}} f(\overrightarrow{xy}) = \sum_{\substack{x \notin S \\ y \in S \\ \overrightarrow{xy} \in E}} 0 = 0$$

Luego junto con la anterior propiedad tenemos que:

$$v(f) = f(S, \overline{S}) - f(\overline{S}, S) = \mathrm{CAP}(S) - 0 = \mathrm{CAP}(S)$$

$$\therefore \quad v(f) = \mathrm{CAP}(S)$$

**Vuelta** ( $\Longleftarrow$ ):

Supongamos $\exists\ S$ corte $v(f) = \mathrm{CAP}(S)$.

Sea $g$ flujo. Por **(1)** $v(g) \leq \mathrm{CAP}(S) \overset{Hipotesis}{=} v(f)$

$\implies v(g) \leq v(f)$

$\therefore \quad f$ es maximal.

$\square$

*Dem.* Por último demostremos **(3)**:

Supongamos $T$ corte. Luego por **(1)**

$\mathrm{CAP}(T) \geq v(f) \overset{Hipotesis}{=} \mathrm{CAP}(S)$

$\implies \mathrm{CAP}(T) \geq \mathrm{CAP}(S)$

$\therefore$ S es minimal. $\square$

# Pregunta 8: Teorema de Brooks

## 3.1 Enunciado detallado

Probar que si G es conexo no regular, entonces $X(G) \leq \Delta(G)$

## 3.2 Resolución

*Dem.* Como G es no regular $\Longrightarrow \delta(G) < \Delta(G)$

Sea $x \in V \mid d(x) = \delta(G)$ y sea $x = v_0, v_1, \ldots, v_r$ el orden de vértices obtenido por correr BFS a partir de $x$. Como $G$ es conexo, BFS agrega todos los vértices de G en cierto orden.

Como el orden se obtiene de BFS, todo vértice distinto de $x$ es agregado por uno de sus vecinos $\Longrightarrow$ **todo vértice distinto de $x$ tiene un vecino posicionado antes de sí en el orden**.

Sea ahora un nuevo orden igual a invertir el anterior: $v_r, v_{r-1}, \ldots, v_1, v_0 = x$
En este nuevo orden, **todo vértice distinto de $x$ tiene un vecino posicionado después de sí en el orden**.

Probemos que al correr *Greedy* en este orden invertido se obtiene un coloreo con a lo sumo $\Delta(G)$ colores.

**Caso 1:** Coloreo de $z \in V \mid z \neq x$

En el peor caso, este vértice tiene todos sus vecinos menos uno antes en el orden, todos con colores distintos, por lo que Greedy va a eliminar $d(z) - 1 \leq \Delta(G) - 1$ colores, por lo que se puede colorear con $\Delta(G)$ colores.

**Caso 2:** Coloreo de $x$

En el peor caso, todos los vecinos de $x$ fueron coloreados con distintos colores. Como la cantidad de vecinos de $x$ es $d(x) = \delta(G) < \Delta(G) \Longrightarrow$ Lo puedo colorear con $\Delta(G)$ colores.

En ambos casos pudimos colorear con $\Delta(G)$ colores, y como Greedy produce un coloreo propio $\Longrightarrow X(G) \leq \Delta(G)$

□

# Pregunta 9: Complejidad 2-Color

## 4.1 Enunciado detallado

Probar que 2-COLOR es polinomial.

## 4.2 Resolución

La prueba basta darla para grafos conexos, ya que si podemos resolverlo en grafos conexos, extenderlo a grafos no conexos es simplemente correr la solución en cada subgrafo conexo.

Sea $G = (V, E)$ grafo conexo. Sea $x \in V$.

> *Algoritmo 2-color:*
> Corro el algoritmo BFS empezando en x.
> Sea $N(z)$ = nivel $z$ en el arbol BFS, es decir la distancia entre $z$ y $x$ en el árbol.
> Sea $c(z) = n(z) \mod 2$
> Luego $c$ es propio $\iff$ $G$ es 2-colorable

Tenemos dos proposiciones:

1. El algoritmo es polinomial.
2. El coloreo $c$ del algoritmo es propio $\iff$ $G$ es 2-colorable

Si probamos **1** y **2** entonces probamos que 2-Color es polinomial.

*Dem.* Probemos **(1)**

BFS es $\mathcal{O}(m)$ y chequear que el coloreo sea propio se puede hacer mientras se corre BFS, por lo que también es $\mathcal{O}(m) \implies$ el algorimo es $\mathcal{O}(m)$

$\therefore$ El algoritmo es polinomial. $\square$

*Dem.* Probemos **(2)** $\equiv c$ es propio $\iff G$ es 2-colorable

**Ida:** ( $\implies$ ) Trivial ya que el algoritmo provee el coloreo.

**Vuelta:** ( $\impliedby$ ) **Hipótesis**: G es 2-coloreable.

*Supongamos* que $c$ que no es propio.

$\implies \exists\ v, z \mid c(v) = c(z) \land vz \in E$

Con $c$ siendo la función de coloreo del algoritmo.

$\implies N(v) = N(z) \quad \mod 2$

Sea $dbfs(a, b)$ la *distancia* entre $a$ y $b$ en el arbol generado al correr BFS a partir de $x$.

Se define la 'distancia' entre dos vértices como la longitud mínima del camino que empieza en uno de esos vértices y termina en el otro

Como el algoritmo es BFS, tenemos las siguientes propiedades:

- $\exists\ dbfs(a, b) \quad \forall\ a, b \in V$
- $dbfs(a, b) = dbfs(b, a) \quad \forall a, b \in V$
- $dbfs(x, y)$ es igual a la distancia de $x$ a $y$ en el grafo $G \quad \forall y \in V$.
- $dbfs(x, y) = N(y) \quad \forall y \in V$ por la definición de Nivel en un arbol

Luego: $c(v) = c(z) \implies dbfs(x, v) \equiv dbfs(x, z)\ _{\text{mod } 2}$

Es decir ambas distancias son pares o impares

Tomemos el camino entre $x$ y $v$ en BFS y el camino entre $x$ y $z$ en BFS. Sea $w$ el último vértice en común entre esos dos caminos. $w$ existe ya que por lo menos $x$ es un vértice en común.

Como $vz \in E$, entonces sea el siguiente ciclo en $G$: $w \dots vz \dots w$

Sea la longitud de este ciclo $L$.

$$L = 1 + dbfs(v, w) + dbfs(z, w)$$
$$\implies L\ _{\text{mod } 2} = 1 + dbfs(v, w) + dbfs(z, w)\ _{\text{mod } 2}$$

Como $\quad 2 * dbfs(x, w)\ _{\text{mod } 2} = 0$

$$\implies L\ _{\text{mod } 2} = 1 + dbfs(v, w) + dbfs(z, w) + 2 * dbfs(x, w)\ _{\text{mod } 2}$$
$$\equiv L\ _{\text{mod } 2} = 1 + [dbfs(v, w) + dbfs(x, w)] + [dbfs(z, w) + dbfs(x, w)]\ _{\text{mod } 2}$$

Pero por construcción de $w$:

$\implies dbfs(v,w) + dbfs(x,w) = dbfs(x,v)$ y también: $dbfs(z,w) + dbfs(x,w) = dbfs(x,z)$

$$\implies L \;_{\text{mod } 2} = 1 + dbfs(x,v) + dbfs(x,z) \;_{\text{mod } 2}$$

Y como $dbfs(x,y) = N(y) \quad \forall y \in V$ y por definicion de $c$:

$[dbfs(x,v) \equiv c(v)]\;_{\text{mod } 2} \quad \wedge \quad [dbfs(x,z) \equiv c(z)]\;_{\text{mod } 2}$

$$\implies L \;_{\text{mod } 2} = [1 + c(v) + c(z)]\;_{\text{mod } 2}$$

Como ambas eran pares o impares, $\implies [c(v) + c(z) \equiv 0]\;_{\text{mod } 2}$

Ya que si ambas eran impares: $c(v) = 1 = c(z) \implies [1 + 1 \equiv 0]\;_{\text{mod } 2}$

Y si ambas eran pares: $c(v) = 0 = c(z) \implies [0 + 0 \equiv 0]\;_{\text{mod } 2}$

$$\therefore \quad L \;_{\text{mod } 2} = 1$$

Por lo tanto el ciclo es **Impar**. Luego como el grafo tiene un subgrafo ciclo impar dentro, $\implies X(G) = 3 \implies$ No existe solución con 2 colores.

**Absurdo** pues por **Hipótesis** G es 2-coloreable.

Luego nuestra suposición inicial es incorrecta:

$\therefore \quad c$ es propio.

$\square$

# Pregunta 10: Teorema de Hall

## 5.1 Enunciado detallado:

Enunciar y probar el Teorema de Hall

## 5.2 Conocimiento presupuesto para la resolución

### 5.2.1 Definiciones

*Def:* **Grafo Bipartito**
Un grafo $G$ es *bipartito* si $X(G) = 2$

*Def:* **Vecindario de un Conjunto de Vértices**
Si $W \subseteq V$, entonces: $\Gamma(W) = \{v \mid \exists\ w \in W : wv \in E\} = \bigcup_{w \in W} \Gamma(w)$

*Def:* **Matching**
Un matching en un grafo $G$ es un **subgrafo** M con $d_M(x) = 1 \quad \forall x \in V(M)$

*Def:* **Partes de un Matching**
En un grafo bipartito definimos las partes $X$ e $Y$ y definimos el grafo como $G = (X \cup Y, E)$
de tal manera que:

$$X \cap Y = \varnothing \ \wedge \ X \cup Y = V \quad \wedge \quad \nexists\ xy \in E \mid x, y \in X \ \vee \ x, y \in Y$$

En otras palabras decimos que $X$ e $Y$ son las partes de un grafo $G$ bipartito si $X$ e $Y$ están separados de tal forma que no hay lados entre elementos de $X$ ni entre elementos de $Y$
Una forma de obtener estas partes cuando se tiene un grafo bipartito es correr el algoritmo de 2-color y definir que los que se coloreen con un color pertenecen a $X$ y los que se coloreen con el otro pertenecen a $Y$

*Def:* **Matching Perfecto**
Un matching es perfecto si $V_M = V(G)$

*Def:* **Matching Completo**
Si $G = (X \cup Y, E)$ es bipartito, un matching is completo de $X$ a $Y$ si $V_M \cap X = X$

### 5.2.2 Flujo maximal para matchings

*Def:* **Problema Matching Maximal**
Dado $G$ bipartito, hallar un matching en G con la mayor cantidad de lados posibles.

Para resolver este problema, lo podemos mapear como un problema de flujo maximal, para poder reutilizar las soluciones de maxflow.

El mapeo es el siguiente:

Dado $G$ grafo bipartito con partes $X$ e $Y$. construimos el siguiente network:

Vértices: $\{s, t\} \cup X \cup Y$

Lados: $\{\overrightarrow{xy} \mid x \in X \wedge y \in Y \wedge xy \in E\} \cup \{\overrightarrow{sx} \mid x \in X\} \cup \{\overrightarrow{yt} \mid y \in Y\}$

Capacidades: Las capacidades de **todos** los lados es 1

*Propiedad:*
Flujos enteros maximales en estos Networks construidos se corresponden con matching maximales en G

*Propiedades:* **Dado un $f$ flujo entero sobre estas networks**
Como la capacidad de todo lado es 1, se cumple que:

$$\mathrm{IN}_f(v) = 0 \quad \vee \quad \mathrm{IN}_f(v) = 1 \quad \forall v \neq s, t, v \in V$$

Además:

$$\mathrm{IN}_f(v) = 0 \iff \mathrm{OUT}_f(v) = 0 \quad \forall v \neq s, t, v \in V$$
$$\mathrm{IN}_f(v) = 1 \iff \mathrm{OUT}_f(v) = 1 \quad \forall v \neq s, t, v \in V$$

Por último, como consecuencia directa de lo anterior:
Dado $x \in X$. $\quad y, z \in Y$ entonces:

$$f(\overrightarrow{xy}) = 1 = f(\overrightarrow{xz}) \implies y = z$$
$$f(\overrightarrow{xy}) = 1 = f(\overrightarrow{zy}) \implies x = z$$

## 5.3 Resolución

> *Teo:* **Teorema de Hall**
> Si $G = (X \cup Y, E)$ es bipartito entonces:
>
> $$\exists \text{ Matching completo de } X \text{ a } Y \iff |S| \leq |\Gamma(S)| \ \forall S \subseteq X \text{ (Condición de Hall)}$$

*Dem.* **Ida** ( $\Longrightarrow$ )

Si existe matching completo de $X$ a $Y$, existe función inyectiva de $X$ a $Y$.

Sea esta función $\phi : X \to Y$ tal que $x\phi(x) \in E \ \wedge \ \phi(x) \in Y \ \forall x \in X$

Sea $S \subseteq X$ y sea $\phi(S) = \{\phi(s) \mid s \in S\} \Longrightarrow |\phi(S)| = |S| \quad \textbf{(1)}$

Sea $p \in \phi(S) \overset{\text{Def } \phi}{\Longrightarrow} \exists\ x \in S \subseteq X \mid xp \in E \Longrightarrow p \in \Gamma(S)$

$\therefore \quad \phi(S) \subseteq \Gamma(S) \Longrightarrow |\phi(S)| \leq |\Gamma(S)| \overset{\textbf{(1)}}{\Longrightarrow} |S| = |\phi(S)| \leq |\Gamma(S)|$

$\therefore \quad |S| \leq |\Gamma(S)|$

**Vuelta** ( $\Longleftarrow$ )

**NOTA:** En esta parte de la demonstración se utilizarán varias propiedades del algoritmo para encontrar matching maximales. Este algoritmo y sus propiedades están descritos más abajo en la sección 5.2.2 (pg. 15)

Lo probaremos por la **contrarecíproca**, es decir:

$$\nexists \text{ Matching completo de } X \text{ a } Y \Longrightarrow \exists\ S \subseteq X \text{ tal que } |S| > |\Gamma(S)|$$

**Hipótesis:** $\nexists$ Matching completo de $X$ a $Y$

Esto implica que **si** usamos el algoritmo para encontrar flujos maximales, vamos a encontrar un matching maximal que **no cubre a X**.

Sea $f$ el flujo maximal obtenido por este algoritmo. Luego como no existe matching completo, $v(f) < |X|$. A su vez sea $C$ el corte minimal obtenido también del algoritmo maximal.

Sea $S = C \cap X$.

Sea $T = C \cap Y$

Vamos a probar que $T = \Gamma(S)$:

Primero probemos que $T$ está contenido en $\Gamma(S)$:

Sea $t \in T$. Como $T \subseteq C$, $t$ forma parte de la última cola del algoritmo $\implies$ fue agregado por algún vecino. Sea este vecino $x$.

Como el grafo es bipartito, y $T \subseteq Y$, $x \in X$. A su vez, $x$ también está en la cola $\implies x \in C$

$x \in X \land x \in C \implies x \in S \overset{\text{x vecino de t}}{\implies} t \in \Gamma(x) \subseteq \Gamma(S) \implies t \in \Gamma(S)$

Esto se cumple para todo $t \in T$

$\therefore \quad T \subseteq \Gamma(S) \qquad$ **(1)**

Ahora probemos que $\Gamma(S)$ está contenido en $T$:

Sea $y \in \Gamma(S) \implies \exists\ x \in S \mid xy \in E$

Como el grafo es bipartito, y $xy \in E$ entonces $y \in Y$. Si probamos que $y \in C$ entonces probamos que $y \in T$

Hay dos casos respecto a como queda el lado $\overrightarrow{xy}$ en el flujo maximal $f$:

*Caso 1:* $f(\overrightarrow{xy}) = 0 \implies x$ agrega a $y$ a la cola $\implies y \in C$

*Caso 2:* $f(\overrightarrow{xy}) = 1$

Como $x \in S \implies x \in C \implies$ algún vértice $z$ agregó a $x$ a la cola.

Como $f(\overrightarrow{xy}) = 1 \implies \mathrm{OUT}_f(x) = 1 \overset{f \text{ es flujo}}{\implies} \mathrm{IN}_f(x) = 1 \implies f(\overrightarrow{sx}) = 1$

$\implies s$ no agregó a $x$ a la cola $\implies z \neq s$

Como $z$ debe ser vecino de $x$ y $z \neq s \implies z \in Y$

Para agregar a $x$, lo debe haber hecho de forma *backward* $\implies f(\overrightarrow{xz}) = 1$

Pero $f(\overrightarrow{xz}) = 1 \ \land \ f(\overrightarrow{xy}) = 1 \implies y = z \implies$ y estaba en la cola $\implies y \in C$

Luego en ambos casos tenemos que $y \in C \implies y \in T$

En resumen probamos que: $y \in \Gamma(S) \implies y \in T$

$\therefore \quad \Gamma(S) \subseteq T$ **(2)**

**(1)** y **(2)** $\implies T = \Gamma(S) \qquad$ **(3)**

Sea $S_0 = \{x \in X \mid \mathrm{IN}_f(x) = 0\}$

Por construcción $s$ agrega a los vértices de $S_0$ a la cola

$\therefore \quad S_0 \subseteq S \qquad$ **(4)**

Como estamos suponiendo que $v(f) \neq |X| \implies$ existe $S_0 \neq \varnothing \implies |S| > 0 \qquad$ **(5)**

Queremos comparar $S - S_0$ con $T$

$y \in T \iff y$ es puesto en la cola por alguien $(y \in Y)$, pero no puede poner a $t$ en la cola $(y \in C)$.

$\implies f(\overrightarrow{yt}) = 1 \implies \mathrm{OUT}_f(y) = 1 \implies \mathrm{IN}_f(y) = 1$

$\implies \exists\ x \mid f(\overrightarrow{xy}) = 1 \implies y$ agrega a $x$ a la cola *backward* $\implies x \in S$

Tambien $f(\overrightarrow{xy}) = 1 \implies \mathrm{OUT}_f(x) = 1 \implies \mathrm{IN}_f(x) = 1 \implies x \notin S_0$

$x \in S\ \wedge\ x \notin S_0 \implies x \in S - S_0$

En resumen, dado $y \in T \implies \exists\ x \in S - S_0 \mid f(\overrightarrow{xy}) = 1$

Por propiedad de flujo, si $f(\overrightarrow{zy}) = 1 \implies z = x$, es decir que hay **solo un** $x$ con $f(\overrightarrow{xy}) = 1$

$\implies$ Existe una función **inyectiva** $Y \to X$ de $T$ a $S - S_0$

Y por un análisis similar puedo ver que si tomo $x \in S - S_0$ se cumple que:

$x \in S - S_0 \implies \mathrm{IN}_f(x) = 1 \implies \mathrm{OUT}_f(x) = 1 \implies \exists\ y \mid f(\overrightarrow{xy}) = 1$

$\implies y \in \Gamma(S) = T \implies$ tengo una **biyección** entre $T$ y $S - S_0$

Como tengo una biyección, esto significa que $|T| = |S - S_0|$ **(6)**

Finalmente:

$|\Gamma(S)| \overset{(3)}{=} |T| \overset{(6)}{=} |S - S_0| = |S| - |S_0| \overset{(5)}{<} |S|$

$\therefore\ \ |\Gamma(S)| < |S|$

Esto prueba la contrarecíproca:

$\therefore\ \ |S| \leq |\Gamma(S)|\ \forall S \subseteq X \implies \exists$ Matching completo de $X$ a $Y$

$\square$

# Pregunta 11: Teorema de König

## 6.1 Enunciado detallado

Enunciar y probar el Teorema del matrimonio de König

## 6.2 Resolución

Todo grafo bipartito regular tiene un matching perfecto.

Dado $W \subseteq V$, definimos
$E_W = \{zw \in E \mid w \in W\}$ Todos los lados que tengan algun vértice en el conjunto.

*Dem.* Sea $G = (X \cup Y, E)$ bipartito

Primero probemos una propiedad de $E_W$:

Sea $W \subseteq X \vee W \subseteq Y$

Como $G$ es bipartito $\implies X(\text{G}){=}\ 2 \implies E \neq \varnothing \implies 0 < \Delta$

Como $G$ es regular $\implies \delta = \Delta \implies d(z) = \Delta \quad \forall\ z \in V$

$\therefore \quad d(z) = \Delta > 0 \quad \forall\ z \in V \quad \mathbf{(1)}$

Por definición: $|E_W| = |\{zw \in E \mid w \in W\}|$

Como no hay lados entre vértices de $X$ o entre vértices de $Y$, tanto en el caso que $W \subseteq X$ como que $W \subseteq Y$, se cumple que cada lado que se agrega a $E_W$ solo una vez.

$$\implies |E_W| = \sum_{w \in W} |\Gamma(w)| \overset{\text{Def } d}{=} \sum_{w \in W} d(w) \overset{(1)}{=} \Delta |W|$$

En conclusión:

$$W \subseteq X \vee W \subseteq Y \implies |E_W| = \Delta |W| \quad \mathbf{(2)}$$

**Ahora probemos que todo grafo bipartito regular tiene un matching completo:**

Sea $S \subseteq X$ y sea $xy \in E_S \mid x \in S, y \in Y$

Por definición: $E_{\Gamma(S)} = \{zw \in E \mid w \in \Gamma(S)\}$

Como $x \in S \implies y \in \Gamma(S) \implies xy \in E_{\Gamma(S)}$

$\therefore \quad E_S \subseteq E_{\Gamma(S)} \implies |E_S| \leq |E_{\Gamma(S)}|$

Como $S \subseteq X \overset{\textbf{(2)}}{\implies} |E_S| = \Delta|S|$

Como $\Gamma(S) \subseteq Y \overset{\textbf{(2)}}{\implies} |E_{\Gamma(S)}| = \Delta|\Gamma(S)|$

Entonces: $|E_S| \leq |E_{\Gamma(S)}| \implies \Delta|S| \leq \Delta|\Gamma(S)| \implies |S| \leq |\Gamma(S)|$

Lo unico que supusimos de $S$ es que $S \in X$:

$\therefore \quad |S| \leq |\Gamma(S)| \; \forall S \subseteq X$ **Condición de Hall**

Por *Teorema de Hall* $\implies \exists$ Matching completo de $X$ a $Y$

**Probemos que este matching es *perfecto***

Para eso probemos que $E_Y = E_X$

Como el $G$ es bipartito, todo lado va a tener un vértice en $X$ y un vértice en $Y$, luego todo lado de $E$ va a ser agregado a $E_X$ y también a $E_Y$

$\therefore \quad E = E_Y = E_X \implies |E_Y| = |E_X| \overset{\textbf{(2)}}{\implies} \Delta|Y| = \Delta|X|$

$\therefore \quad |Y| = |X|$

*Supongamos* ahora que el matching **no** es perfecto $\implies V_M \neq V$

Como siempre se cumple que $V_M \subseteq V \overset{V_M \neq V}{\implies} V_M \subset V \implies \exists \; v \in V \mid v \notin V_M$

También se cumple siempre que $X \cup Y = V \implies v \in X \; \vee v \in Y$

*Caso 1:* $v \in X$ Pero como el matching es completo $X$ a $Y \implies V_M \cap X = X$

$\implies x \in V_M \quad \forall \; x \in X \implies v \in V_M$ **Absurdo.**

*Caso 2:* $v \in Y \quad v \notin V_M \implies$ Existe un vértice en $Y$ que **no** pertenece al matching. Como el matching es completo $X$ a $Y$, todo vértice de $X$ tiene un vértice $y \in Y$ que sí pertenece $\implies |Y| > |X|$

Pero $|Y| = |X|$ **Absurdo.**

Luego nuestra suposición está equivocada:

$\therefore \quad$ El matching es perfecto. $\square$

# Pregunta 14: Teorema de la cota de Hamming

## 7.1 Enunciado detallado

Enunciar el teorema de la cota de Hamming y probarlo

## 7.2 Conocimiento presupuesto para la resolución

*Def:* **Códigos de Corrección**
Un **código** $C$ es un conjunto $\neq \varnothing$ de palabras sobre un alfabeto $A$

*Def:* **Código de Bloque**
También llamado *block-code*, es un código en donde todas las palabras tienen la misma longitud.
$\implies \exists\ n \mid C \in A^n$

*Def:* **Código Binario**
Es un código donde el alfabeto es $\{0,1\}$

*Def:* **Distancia de Hamming**
Dadas dos palabras $v, w \in \{0,1\}^n$, la distancia entre v, w es:

$$d(v,w) = d_H(v,w) = \text{Cantidad de bits de diferencia entre } v \text{ y } w$$

*Def:* $\delta$
Dado un código $C$: $\quad \delta(C) = Min\{d_H(v,w) \mid v, w \in C, v \neq w\}$

*Propiedad:* **La distancia de Hamming es una distancia**

Se cumples las propiedades matemáticas de las distancias:
$\forall\ v, w, u \in \{0,1\}^n$ palabras:

1. $d_H(v,w) = d_H(w,v)$
2. $d_H(v,w) \geq 0$
3. $d_H(v,w) = 0 \iff v = w$
4. $d_H(v,w) \leq d_H(v,u) + d_H(u,w)$ (Desigualdad triangular)

*Def:* **Disco de radio**

Dada una palabra $v \in \{0,1\}^n$, y un número natural $r \geq 0$, definimos el **disco de radio** $r$ **alrededor de** $v$ como:

$$D_r(v) = \{w \in \{0,1\}^n \mid d_H(v,w) \leq r\}$$

*Def:* **Detección**

Un código C **detecta** $r$ errores si $D_r(v) \cap C = \{v\} \quad \forall\ v \in C$

*Def:* **Corrección**

Un código C **corrige** $r$ errores si $D_r(v) \cap D_r(w) = \varnothing \quad \forall\ v, w \in C \mid v \neq w$

*Teo:*

Sea $C$ un código y $\delta = \delta(C)$. Entonces:

- $C$ detecta $\delta - 1$ errores, pero no detecta $\delta$
- Si $t = \lfloor \frac{\delta-1}{2} \rfloor$ entonces C corrige $t$ errores pero no corrige $t+1$

## 7.3 Resolución

*Teo:* **Cota de Hamming**

Sea $C$ un código de longitud $n$, $\delta = \delta(C)$, $t = \lfloor \frac{\delta-1}{2} \rfloor$. Entonces:

$$|C| \leq \frac{2^n}{\sum_{r=0}^{t} \binom{n}{r}} = \frac{2^n}{1 + n + \cdots + \binom{n}{t}}$$

*Dem.* Sea $A = \bigcup_{v \in C} D_t(v)$

Por teorema sabemos que $C$ corrige $t$ errores $\implies D_t(v) \cap D_t(w) = \varnothing\ \forall\ w, v \in C$

$\implies$ A es formada por una **unión disjunta**:

$\implies |A| = \sum_{v \in C} |D_t(v)|$ **(1)**

Solo falta descubrir la cardinalidad de los $D_t$

Sea $S_r(v) = \{w \in \{0,1\}^n \mid d_H(v,w) = r\}$

Por definición de $D_t(v)$, tenemos que:

$$D_t(v) = \bigcup_{r=1}^{t} S_r(v)$$

Como la distancia entre dos vértices es única, esta union también es disjunta.

$$\Longrightarrow |D_t(v)| = \sum_{r=0}^{t} |S_r(v)| \quad \mathbf{(2)}$$

Falta ver la cardinalidad de cada $S_r(v)$. Cada elemento del conjunto es una palabra que difiere de $v$ en exactamente $r$ de los $n$ lugares posibles. Como el Código es binario, que $w$ varíe de $v$ en el lugar $i$ es significa que $w_i = 1 \oplus v_i$, es decir solo hay una forma de que varíe en ese bit. Es decir que la cardinalidad de $S_r(v)$ es las distintas formas de tomar $r$ espacios de $n$ posibles. Esto es el número combinatorio $\binom{n}{r}$

$$\Longrightarrow |S_r(v)| = \tbinom{n}{r} \quad \text{Juntando eso con } \mathbf{(2)}\text{:}$$

$$\Longrightarrow |D_t(v)| = \sum_{r=0}^{t} \binom{n}{r} \quad \text{Y juntando esto con } \mathbf{(1)}\text{:}$$

$$\Longrightarrow |A| = \sum_{v \in C} \left[ \sum_{r=0}^{t} \binom{n}{r} \right]$$

Como la suma interior no depende de $v$, esta suma va a ocurrir la cantidad de elementos que tenga $C$

$$\Longrightarrow |A| = |C| \left[ \sum_{r=0}^{t} \tbinom{n}{r} \right]$$

$$\Longrightarrow |C| = \frac{|A|}{\sum_{r=0}^{t} \binom{n}{r}}$$

Como $A$ es un subconjunto de $\{0,1\}^n \Longrightarrow |A| \leq |\{0,1\}^n| \quad (3)$

Cada elemento de $\{0,1\}^n$ tiene $n$ letras que pueden ser 0 o 1. Con 2 opciones por cada letra, y cada palabra siendo de largo $n$, existen $2^n$ palabras distintas.

Luego $|\{0,1\}^n| = 2^n \overset{(3)}{\Longrightarrow} |A| \leq 2^n$

$$\therefore \quad |C| \leq \frac{2^n}{\sum_{r=0}^{t} \binom{n}{r}} \qquad \square$$

# Pregunta 15: Relación entre $H$ y $\delta(C)$

## 8.1 Enunciado detallado

Probar que si $H$ es matriz de chequeo de $C$, entonces:

$$\delta(C) = \min\{j \mid \exists \text{ un conjunto de } j \text{ columnas LD de } H\}$$

## 8.2 Conocimiento presupuesto para la resolución

*Def:* **Cuerpo**

Un cuerpo es un conjunto $K$ y dos operaciones $(+)$ y $(\cdot)$, llamadas *suma* y *multiplicaciones* respectivamente, con las siguientes propiedades:
$\forall \quad a, b \in K$ se cumple que $a + b \in K \ \wedge \ a \cdot b \in K$
La adición y multiplicacion son *asociativas* y *conmutativas*.
Existe un elemento *neutro* para la suma y para la multiplicación (0 y 1)
Para cada elemento existe un opuesto en la suma y un inverso en la multiplicación tal que: $a + (-a) = 0 \quad \wedge \quad a \cdot a^{-1} = 1$
Existe *distributividad* de la multiplicación respecto a la adición: $a \cdot (b + c) = a \cdot b + a \cdot c$

*Def:* **Espacio Vectorial**

Un espacio vectorial sobre un cuerpo $K$ es un conjunto $V$ no vacío y dos operaciones:
Suma: $(+) : V \times V \to V$
Producto: $(\cdot) : K \times V \to V$ (llamado tambien producto escalar)
Con las siguientes propiedades:
La suma y el producto por escalar son *cerradas* y *asociativas* respecto a $V$
Existen elementos *neutros* para la suma y el producto
La suma es *conmutativa* y tiene un *opuesto*.
Hay *distributividad* respecto a la suma vectorial y a la suma escalar: $a \cdot (u + v) = a \cdot u + a \cdot v \ \wedge \ (a + b) \cdot u = a \cdot u + b \cdot u \quad \forall \ a, b \in K, u, v \in V$

*Observación:* $\{0,1\}^n$ **es espacio vectorial**

$\{0,1\}^n$ es un espacio vectorial sobre el cuerpo $\{0,1\}$ con la suma y el producto **módulo 2**

*Def:* **Subespacio Vectorial**

$W$ es un subespacio vectorial de $V$ si:
$W \neq \varnothing$
$\alpha, \beta \in W \implies \alpha + \beta \in W$
$\alpha \in W, c \in K \implies c \cdot \alpha \in W$

*Observación:* **Subespacio en $\{0,1\}^n$**

$C$ es un subespacio vectorial de $\{0,1\}^n \iff (\alpha, \beta \in C \implies \alpha + \beta \in C) \quad \wedge \ C \neq \varnothing$
Es decir que para el caso binario es suficiente probar que la suma es cerrada

*Def:* **Linealmente Independiente**

Un conjunto de vectores es linealmente independiente si ninguno de ellos puede ser escrito como una **combinacón lineal** de los restantes.
Es decir: $\nexists b \in B \mid \exists\ a_1, \ldots a_n \in K, v_1, \ldots v_n \in B \ : \ a_1 \cdot v_1 + \cdots + a_n \cdot v_n = b$

*Def:* **Base de un Espacio Vectorial**

Una base $B$ de un espacio vectorial $V$ sobre un cuerpo $K$ es un subconjunto **linealmente independiente** de $V$ que puede generar cualquier elemento de $V$.
Es decir que para cualquier vértice $v \in V$, existe una **combinación lineal** de los elementos de $B$ que igualan a $v$

*Def:* **Dimensión de un Espacio Vectorial**

Es la **cardinalidad** de cualquiera de sus bases.

*Def:* **Transformaciones Lineales**

Una transformación lineal entre espacios vectoriales $V_1$ y $V_2$ es una **función** $T : V_1 \to V_2$ tal que:
$\alpha, \beta \in V_1, k \in K \implies T(k \cdot \alpha + \beta) = k \cdot T(\alpha) + T(\beta)$

*Def:* **Imagen de una transformación lineal**

$\mathrm{Im}(T) = \{\beta \in V_2 \mid \exists\ \alpha \in V_2 \ : \ T(\alpha) = \beta\}$

*Def:* **Nucleo de una transformación lineal**

Dado $T$ transformación lineal, el núcleo de $T$ es:
$\mathrm{Nu}(T) = \{\alpha \in V_1 \mid T(\alpha) = \overrightarrow{0}\}$ donde $\overrightarrow{0}$ es el vector cero de $V_2$

*Propiedad:* **Subespacios de Transformaciones lineales**

Si $T : V_1 \to V_2$ es una transformación lineal, entonces:

- $\mathrm{Nu}(T)$ es un subespacio vectorial de $V_1$
- $\mathrm{Im}(T)$ es un subespacio vectorial de $V_2$

*Def:* **Códigos Lineales**
Un código **lineal** de longitud $n$ es un **subespacio vectorial** de $\{0,1\}^n$

*Def:* **Matriz Generadora**
Diremos que una matriz $G$ que sea $k \times n$ es una **matriz generadora** de un código lineal $C$ si las filas de $G$ forman una base de $C$
Además, $C$ es tiene dimensión $k$ y longitud $n$.

*Def:* **Matriz de Chequeo**
$H$ es una matriz de chequeo de un código $C$ si:
$C = \text{Nu}(H) = \{x \mid Hx^t = 0\}$
Además $H$ es $(n-k \times n)$ si las filas son Linealmente Independientes.

*Def:* **Peso de Hamming**
Dada una palabra $v$ de un código, el peso de Hamming de $v$ es $|v| = d_H(v,0)$.
Es decir el número de unos que tiene $v$

*Observación:*
$v \in C$ código lineal $\implies |v| =$ cantidad de '1' en $v$

*Lema:* **$\delta$ en códigos lineales**
Si $C$ es lineal, entonces $\delta(C) = \min\{|v| \;:\; v \in C, v \neq 0\}$

## 8.3 Resolución

*Teo:* **Relación entre H y $\delta$**
Si $H$ es matriz de chequeo de $C$, entonces:

$$\delta(C) = \min\{j \mid \exists \text{ un conjunto de } j \text{ columnas LD de } H\}$$

LD: Linealmente Dependientes

*Dem.* Sea $H_j$ la columna *j-esima* de $H$.

Sea $\{H_{j_1}, H_{j_2}, \dots, H_{j_s}\}$ un conjunto **LD** de columnas de $H$

$\implies \exists\ c_1, c_2, \dots, c_s \mid c_1 H_{j_1} + c_2 H_{j_2} + \cdots + c_s H_{j_s} = 0 \ \wedge [\exists\ 0 < i \leq s : c_i \neq 0]$ **(1)**

Sea $e_i$ el vector con todos 0 salvo un 1 en la coordenada $i$.

Por definición de producto de matrices se cumple que: $He_{j_i}{}^t = H_i$ **(2)**

Luego sea $w = c_1 e_{j_1} + c_2 e_{j_2} + \cdots + c_s e_{j_s}$. Entonces:

$Hw^t = H\ (c_1 e_{j_1} + c_2 e_{j_2} + \cdots + c_s e_{j_s})^t$

Luego por distributividad de matrices:

$Hw^t = c_1 H e_{j_1}{}^t + c_2 H e_{j_2}{}^t + \cdots + c_s H e_{j_s}{}^t$

$\overset{\mathbf{(2)}}{\Longrightarrow}\ Hw^t = c_1 H_{j_1} + c_2 H_{j_2} + \cdots + c_s H_{j_s} \overset{\mathbf{(1)}}{\Longrightarrow} = 0$

Por definición de *nucleo* $\Longrightarrow\ w \in \mathrm{Nu}(H)$

Y como $H$ es matriz de chequeo (def de matriz de chequeo) se cumple que $C = \mathrm{Nu}(H)$

$\Longrightarrow\ w \in C$

Ahora veamos $|w|$. Como $w$ es suma de a lo sumo $s$ '$e_i$' y cada $e_i$ tiene un 1, entonces $|w| \leq s$

Luego por Lema: $\delta(C) = \min\{|v|\ :\ v \in C, v \neq 0\}$

$\Longrightarrow\ \delta(C) = \min\{|v|\ :\ v \in C, v \neq 0\} \leq |w| \leq s$

Esto se cumple para cualquier cualquier conjunto de $s$ columnas LD de $H$

En particular se va a cumplir para el conjunto de menos elementos, es decir:

$\therefore\quad \delta(C) \leq \min\{j \mid \exists \text{ un conjunto de } j \text{ columnas LD de } H\}\quad \mathbf{(3)}$

Ahora probemos el otro lado de la desigualdad:

Por el mismo lema que antes sabemos que $\exists\ v \in C \mid \delta(C) = |v|$

$\Longrightarrow\ v$ tiene $\delta(C)$ '1'

$\Longrightarrow\ \exists\ i_1, i_2, \ldots, i_{\delta(C)} \mid v = e_{i_1} + e_{i_2} + \cdots + e_{i_{\delta(C)}}$

Como $H$ es una transformación lineal y $v \in C = Nu(H) \Longrightarrow\ Hv^t = 0$

Por el mismo cálculo que antes concluimos que:

$H_{i_1} + H_{i_2} + \cdots + H_{i_{\delta(C)}} = 0$

Por definición de LD tenemos que el conjunto $\{H_{i_1}, H_{i_2}, \ldots, H_{i_{\delta(C)}}\}$ es LD

Luego $\delta(C) \in \{j \mid \exists \text{ un conjunto de } j \text{ columnas LD de } H\}$

$\therefore\quad \min\{j \mid \exists \text{ un conjunto de } j \text{ columnas LD de } H\} \leq \delta(C)\quad \mathbf{(4)}$

$\mathbf{(3)} \wedge \mathbf{(4)} \Longrightarrow\ \delta(C) = \min\{j \mid \exists \text{ un conjunto de } j \text{ columnas LD de } H\}$

□

# Pregunta 16: Teo Fundamental de los Códigos Cíclicos

## 9.1 Enunciado detallado

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

1. $C$ está formado por los múltiplos de $g(x)$ de grado menor que $n$:

$$C = \{p(x) \mid gr(p) < n \ \wedge \ g(x)|p(x)\}$$

2. $C = \{v(x) \odot g(x) \mid v \text{ es un polinomio cualquiera}\}$

3. $gr(g(x)) = n - k$

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

## 9.2 Conocimiento presupuesto para la resolución

*Notación:*

Las palabras en códigos cíclicos se denotan como $w_0 w_1 \dots w_{n-1}$

*Def:* **Función Rotación**

Si tenemos una palabra $v = v_1 v_2 \dots v_n$

- Una rotación en **1 bit hacia la derecha** es $v_n v_1 \dots v_{n-1}$
- Una rotación en $s$ **bits hacia la derecha** es hacer $s$ veces la rotación en 1 en derecha

Dada una palabra $w$ definimos la rotación como la palabra:
$\text{rot}^p(w)$ como la rotación en $p$ bits hacia la **derecha** de $w$

*Def:* **Código Cíclico**

Un código es **cíclico** si es lineal y la **rotación** de cualquiera de sus palabras es otra palabra del código.

*Observación:*
Un código $C$ es cíclico si $C$ es lineal y $w \in C \implies \text{rot}(w) \in C$

Podemos pensar en palabras de un código como un polinomio:

*Def:* **Polinomios de Códigos binarios**
Dada la palabra $w \in C, w = w_0 w_1 \dots w_{n-1}$
El polinomio de $w$ es:

$$w = \sum_{i=0}^{n-1} w_i x^i$$

*Notación:* **Producto entre polinomios de C**
Dadas dos palabras $v$ y $w$ de longitud $n$, identificadas con los polinomios $v(x)$ y $w(x)$, definimos:

$$v \odot w = v(x)w(x) \mod (1 + x^n)$$

*Propiedad:* **Rotación en polinomios**

$$\text{rot}(w) = w(x) \odot x$$

*Propiedad:* **$\odot$ es absorbente en C**
Sea $C$ un código cíclico, y sea $w \in C$. Entonces:

$$w \odot v \in C \quad \forall v$$

Notar no dice '$v \in C$'. Esto es a proposito. Es cualquier palabra $v$

*Propiedad:*
Si $C$ es lineal, entonces existe **un único** polinomio no nulo en $C$ de **grado mínimo**

*Def:* **Polinomio Generador**
Si $C$ es cíclico, se llama **polinomio generador** al único polinomio no nulo de menor grado, y se lo denota como $g(x)$

## 9.3 Resolución

> *Teo:* **Teorema Fundamental de Códigos Cíclicos**
> Sea $g(x)$ el polinomio generador de un código cíclico $C$ de longitud $n$. Entonces:
> 1. $C$ está formado por los múltiplos de $g(x)$ de grado menor que $n$. Es decir: $C = \{p(x) \mid gr(p) < n \ \wedge \ g(x)|p(x)\}$
> 2. $C = \{v(x) \odot g(x) \mid v \text{ polinomio}\}$
> 3. $gr(g(x)) = n - k$
> 4. $g(x)$ divide a $1 + x^n$

*Dem.* Probemos los primeros dos puntos del teorema:

Sea $C_1 = \{p(x) \mid gr(p) < n \ \wedge \ g(x)|p(x)\}$

y sea $C_2 = \{v(x) \odot g(x) \mid v \text{ polinomio}\}$

Por la propiedad absorbente, tenemos que $C_2 \subseteq C$

Sea $p(x) \in C$

Diviendo $p(x) porg(x)$ obtenemos los polinomos $q(x)$ y $r(x)$ con $gr(r) < gr(g)$ tal que:

$p(x) = q(x)g(x) + r(x)$

Si probamos que $r(x) = 0$, entonces $p(x) \in C_1$

Como $gr(r) < gr(g) < n$ y $p(x) \in C \implies gr(p) < n$

$\implies r(x) = r(x) \mod (1 + x^n) \quad \wedge \quad p(x) = p(x) \mod (1 + x^n)$

$\implies p(x) \mod (1 + x^n) = \left[q(x)g(x) + r(x)\right] \mod (1 + x^n)$

Como $\mod$ es distributiva con la suma:

$\implies p(x) \mod (1 + x^n) = [q(x)g(x) \mod (1 + x^n)] + [r(x) \mod (1 + x^n)]$

Por definición de $\odot$ y que $r(x) = r(x) \mod (1 + x^n)$

$\implies p(x) = q \odot g + r(x)$

Como la suma en estos polinomios es equivalente a XOR se cumple que la suma es su propia inversa:

$p(x) = q \odot g + r(x)$

$\equiv r(x) = q \odot g + p(x)$

Tenemos que $p(x) \in C$, y como $\odot$ es absorbence en $C$ tenemos que $g(x) \in C \implies q \odot g \in C$

Luego $r(x) = q \odot g + p(x) \in C$

Pero $gr(r) < gr(g)$ que es el polinomio **no nulo** de menor grado de $C$

$\therefore \quad r(x) = 0$ y como dijimos antes:

$\implies p(x) \in C_1 \implies C \subseteq C_1$

Sea $p(x) \in C_1 \implies \exists q(x) \mid p(x) = q(x)g(x)$

Si aplicamos $\mod (1 + x^n)$ en ambos lados tenemos:

$p(x) \mod (1 + x^n) = q(x)g(x) \mod (1 + x^n)$

Pero $p(x) \in C_1 \implies gr(p) < n \implies p(x) \mod (1 + x^n) = p(x)$

$p(x) = q(x)g(x) \mod (1 + x^n)$ y por definición de $\odot$

$p(x) = q \odot g$

Luego $p(x) \in C_2$

$\therefore \quad C_1 \subseteq C_2$

En resumen tenemos que:

$C_2 \subseteq C \subseteq C_1 \subseteq C_2$

$\therefore \quad C_2 = C = C_1$ $\square$

*Dem.* Probemos el punto 3 del teorema:

Sea $t$ el grado de $g(x)$

Por la parte **1.**, $p(x) \in C \iff \exists\, q(x) \mid p(x) = q(x)g(x)$

Como el grado de los elementos de $C$ es menor que n, entonces el grado de $q(x)g(x)$ debe ser menor que $n$.

Por lo tanto el grado de $q(x)$ debe ser menoir que $n - t$

Asíque **para cada polinomio de grado menor que $n - t$ corresponde un polinomio de $C$ y viceversa**.

$\implies$ La cardinalidad de $C$ es la cardinalidad del conjunto de polinomios de grado menor que $n - t$ de $C$.

Cada polinomio de grado menor que $n-t$ tiene $n-t$ coeficientes (del grado 0 al grado $n-t-1$), y cada uno de esos coeficientes puede ser 0 o 1, por lo que hay en total:

$2^{n-t}$ polinomios de $C$

Como $C$ es lineal, sabemos que su cardinalidad es $2^k$

$\implies k = n - t$

$\therefore \quad t = n - k$ $\square$

*Dem.* Probemos el punto 4 del teorema:

Dividimos $(1 + x^n)$ por $g(x)$ y obtenemos $q(x), r(x)$ con $gr(r) < gr(g) \mid (1 + x^n) = q(x)g(x) + r(x)$

Por la misma propiedad usada en la primera prueba, se cumple que:

$r(x) = 1 + x^n + q(x)g(x)$

Como $gr(r) < gr(g) < n \implies r(x)r(x) \mod (1 + x^n)$

$\implies r(x) = \left[1 + x^n + q(x)g(x)\right] \mod (1 + x^n)$

Por definición de $\odot$

$\implies r(x) = \left[1 + x^n\right] \mod (1 + x^n) + q \odot g$

y como $1 + x^n \mod (1 + x^n) = 0$

$\implies r(x) = q \odot g$

Luego por propiedad de absorción de $\odot \implies r(x) \in C$

Pero como $gr(r) < gr(g)$, por lo mismo que antes $\implies r = 0$

Luego $g(x)|(1 + x^n)$ En otras palabras $g(x)$ *divide* a $(1 + x^n)$

$\square$