Contents

1 Matchings

1.1 Encontrar matching que minimice costo total

1.1.1 Correr algoritmo hungaro

1. Restar a cada fila de la matriz su mínimo
   - Si tiene un 0 no hace falta
2. Restar a cada columna de la matriz su mínimo
   - Si tiene un 0 no hace falta
3. Buscar matching armando caminos con EK
4. Si no se puede completar el matching:
   - Tomamos:
   $$S = \text{ filas etiquetadas}$$
   $$\Gamma(S) = \text{ columnas etiquetadas}$$
   - Tachamos en la matriz:
     - Filas de S^C, osea, las que no estan en S
     - Columnas de $\Gamma(S)$
   - Buscar el mínimo m entre los elementos que no fueron tachados
   - Si un elemento está tachado:
     - 0 veces: se le resta m
     - 1 vez: no se hace nada
     - 2 veces: se le suma m
   - Volver a 3
5. Si se puede completar
   - Vemos en la matriz original los valores originales para el matching:
     - Es decir, si tenemos que AII es parte del matching, vemos que valor tenía en la matriz original
   - Sumamos todos los valores de todos los caminos y este es la mínima suma de los costos

1.2 De entre todos los matching que minimicen el costo total, hallar uno que minimize el mayor costo

1. Hacer hungaro para encontrar suma mínima de costos, llamemosla SMC
2. Hacer Gross para encontrar matching que minimice el mayor costo, llamemos a este mayor costo MMC
3. Hacer busqueda binaria para los costos $\geq MMC$:
    - Para cada resultado i de la busqueda binaria:
        - Se modifica la matriz para que todos los elementos $> i$ sean ∞
        - Se corre hungaro sobre esta matriz
        - Se verifica que la mínima suma de costos sea igual a SMC
            * Si se cumple: encontramos el matching que buscamos
            * Si no se cumple: seguir con la siguiente iteracion de busqueda binaria

1.3 minimizar el mínimo (CHEQUEAR)

1. Multiplicar matriz por -1 y sumarle el máximo (positivo)
2. Hacer Hungaro

1.4 Maximizar suma (CHEQUEAR)

1. Usar umbral más grande posible
2. Correr hungaro

2 Codigos lineales

2.1 Determinar dimension k de un codigo

2.1.1 En base a matriz generadora G

Contar cantidad de filas de G, y esa será la dimension

2.1.2 En base a matriz chequeadora H

Contar cantidad de columnas de H, y esa será la dimension

2.2 Dar número de palabras de un codigo

Calcular 2^k

2.3 Dar palabras no nulas

2.3.1 Filas G

Las filas de la matriz G son palabras de C

2.3.2 Multiplicar u por G

$$u \cdot G$$

Esto se traduce a sumar todas las columnas que se correspondan a la posicion de los 1s de u

2.3.3 Metodo 1: matriz generadora

1. Obtener matriz generadora G
2. Tomar una palabra cualquiera u
3. Multiplicar $u \cdot G$

El resultado de (3) será una palabra del codigo

2.3.4 Metodo 2: ???

2.4 Calcular $\delta(C)$ (CHEQUEAR)

2.4.1 Como se calcula delta?

$$\delta(C) = Min\{j : \exists \text{ conjunto de j columnas LD de H } \}$$

2.4.2 Propiedad de matriz chequeadora

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

$\Rightarrow \delta \geq 3$

2.4.3 Pasos

1. Revisar la matriz chequeadora H:
    - Si NO hay una columna igual a 0: $\delta > 1$
    - Si NO hay dos columnas iguales: $\delta > 2$
2. Buscar subconjunto de 3 columnas LD
    - Si se encuentra $\Rightarrow \delta(C) \leq 3$

2.5 Deducir palabra más probable enviada

2.5.1 Multiplicar u por H

$$H \cdot u^t$$

Esto se traduce a sumar todas las filas que se correspondan a la posicion de los 1s de u

2.5.2 Pasos

1. Multiplicar palabra por H, llamemos a este resultado R
2. Si R es igual a la i-esima columna de H:
    - Hacer un bit flip en la i-esima posicion de la palabra
    - El resultado de esto será la palabra más probable que ha sido enviada

3 Codigos ciclicos

3.1 Determinar dimension k de una funcion g

3.1.1 Conociendo el grado gr(g) y la longitud n

Despuejar k en la siguiente ecuacion:

$$gr(g) = n - k$$

3.2 Dar número de palabras de un codigo

Calcular 2^k

3.3 Modulo de polinomios

Hagamos un ejemplo, con $g(x) = (1 + x^2 + x^3)$

$$(x^3 + x^4 + x^6) \mod g(x)$$

3.3.1 Si $i < gr(g)$

$gr(g) = 3$

$$\begin{aligned} 1 &\mod g(x) = 1 \\ x &\mod g(x) = x \\ x^2 &\mod g(x) = x^2 \end{aligned}$$

3.3.2 Si $i = gr(g)$

$$\begin{aligned} &g(x) \mod g(x) = 0 \\ &(1 + x^2 + x^3) \mod g(x) = 0 \\ &x^3 = (1 + x^2 + x^3) - (1 + x^2) \\ &\Rightarrow x^3 \mod g(x) = 1 + x^2 \end{aligned}$$

3.3.3 Si $i > gr(g)$

Se usan los valores previamente calculados:

$$\begin{aligned} x^4 \quad \text{mód } (g(x)) &= xx^3 \quad \text{mód } (g(x)) \\ &= x\left(1 + x^2\right) \quad \text{mód } (g(x)) \\ &= \left(x + x^3\right) \quad \text{mód } (g(x)) \\ &= x + 1 + x^2 \\ &= 1 + x + x^2 \end{aligned}$$

$$
\begin{aligned}
x^5 \mod (g(x)) &= xx^4 \mod (g(x)) \\
&= x\left(1+x+x^2\right) \mod (g(x)) \\
&= \left(x+x^2+x^3\right) \mod (g(x)) \\
&= x+x^2+1+x^2 \\
&= 1+x
\end{aligned}
$$

$$
\begin{aligned}
x^6 \mod (g(x)) &= xx^5 \mod (g(x)) \\
&= x(1+x) \mod (g(x)) \\
&= \left(x+x^2\right) \mod (g(x)) \\
&= x+x^2
\end{aligned}
$$

3.3.4 Resultado

$$
\begin{aligned}
&\left(x^3+x^4+x^6\right) \mod g(x) \\
&= \left(1+x^2\right) \mod g(x) + \left(1+x+x^2\right) \mod g(x) + \left(x+x^2\right) \mod g(x) \\
&= \left(1+x^2\right) + \left(1+x+x^2\right) + \left(x+x^2\right) \\
&= x^2
\end{aligned}
$$

3.4 Codificar palabra

3.4.1 Metodo 1

1. Multiplicar palabra P por el polinomio
2. Simplificar teniendo en cuenta que si tenemos dos miembros iguales estos se cancelan.
    - Es decir, si en el resultado tenemos $x^2 + x^2$, podemos cancelarlos

3.4.2 Metodo 2: ???

Resolver $\left(u(x)x^{n-k} \bmod g(x)\right) + u(x)x^{n-k}$

3.5 Error trapping

Sea $t = \lfloor \frac{\delta-1}{2} \rfloor$ la cantidad de errores máximos que puedo corregir y w la palabra recibida:

1. Calcular sindrome: $s_0 = w \mod g(x)$
2. Ir calculando $s_i = x s_{i-1} \mod g(x)$ hasta que $|S_i| \leq t$
3. Sea $e = x^{n-i} \mod (1 + x^n)$, y v la palabra original

$$|e| \leq t \Rightarrow v = w + e$$

4 Calcular cantidad de codigos (CHEQUEAR)

4.1 Número combinatorio

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

4.2 Codigos lineales

La cantidad de codigos lineales con p palabras y longitud n, es:

$$\frac{(2^n - 1)(2^n - 2)}{(p-1)!}$$

4.2.1 Calcular k (A CHEQUEAR)

Sea p la cantidad de palabras:

$$k = \sqrt{p}$$

4.3 Codigos ciclicos

???