

Algoritmo de Gerrymander - Programación dinámica

por Pablo Valdunciel Sánchez

Algoritmos y Computación 2018/2019

Escuela de Ingeniería Informática

Universidad de Valladolid





Origen del término



Origen del término

1812



Elbridge Gerry → Gerry
← Salamandra → Salamander

Gerry-mander



El problema

CONTEXTO

- Distritos electorales
- Precintos electorales

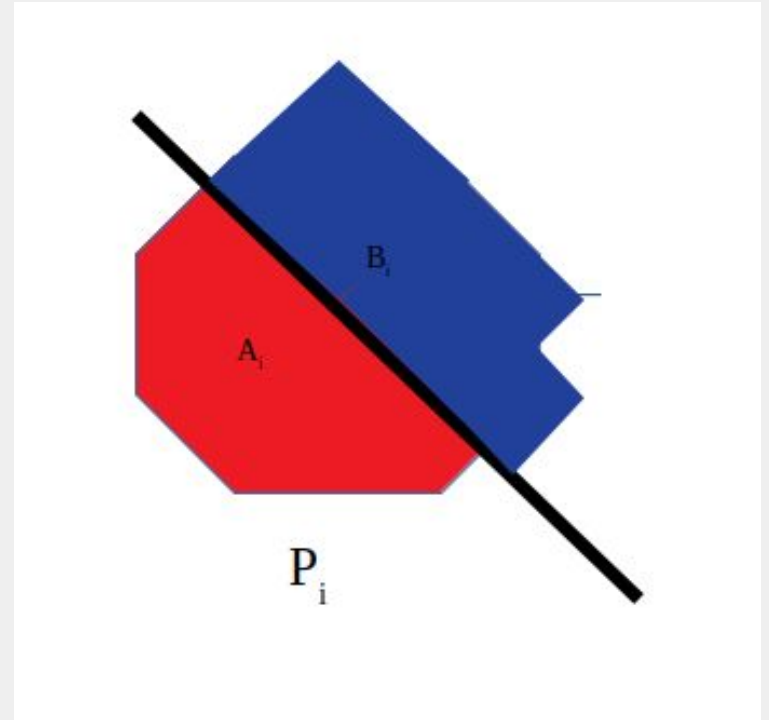
- Un Estado = x distritos
- Un distrito = y precintos
- Distrito Di tiene asignados **ni representantes**
- El partido vencedor en el distrito Di consigue ni representantes, el resto de partidos ninguno.



CONTEXTO

Suponemos :

- + 2 distritos, D1 y D2
- + n precintos (n es PAR)
- + m personas en cada precinto
- + 2 partidos, A ■ y B ■
- + A_i votan al partido A en el precinto P_i (A_1, A_2, \dots, A_n)



- A_i personas votan al partido A ■
- $B_i = m - A_i$ personas votan al partido B ■

→ ENTRADA

1. m : número de personas en cada precinto
2. A_1, A_2, \dots, A_n : votos al partido A en cada precinto (# de precintos implícito)
3. $\text{ady}[1..n][1..n]$: adyacencias entre los precintos

m

A_1, A_2, \dots, A_n

$\text{ady}[1..n][1..n]$

SALIDA →

La salida buscada son dos distritos, **D1** y **D2** que verifiquen las siguientes condiciones:

1. **$|D1| = |D2| = n/2$** - Cada distrito tiene $n/2$ precintos.
2. Los precintos de cada distrito son **CONTIGUOS**.
3. **Mayoría del partido A en D1 y en D2**, es decir,...

$$A(D1) > \frac{(m * n)}{4}$$

$$A(D2) > \frac{(m * n)}{4}$$

Un ejemplo

$n = 4$ precintos
 $m = 80$ personas

$A_1 = 45$

$A_2 = 50$

$A_3 = 34$

$A_4 = 56$

Ady \rightarrow Imagen

| | |
|------------|------------|
| $A_1 = 45$ | $A_2 = 50$ |
| $A_3 = 34$ | $A_4 = 56$ |

$$\text{Mayoría} = (80 \cdot 4) / 4 + 1 = 81$$

Un ejemplo

$$\text{Mayoría} = (80 \cdot 4) / 4 = 80$$

| | |
|------------|------------|
| $A_1 = 45$ | $A_2 = 50$ |
| $A_3 = 34$ | $A_4 = 56$ |

Dos posibilidades que verifquen la condición de contigüidad :

a)

1. **D1: precintos 1 y 3** → **79** votos al partido A → Minoría del partido A en D1
2. **D2: precintos 2 y 4** → **106** votos al partido A → Mayoría del partido A en D2



b)

1. **D1: precintos 1 y 2** → **95** votos al partido A → Mayoría del partido A en D1
2. **D2: precintos 3 y 4** → **90** votos al partido A → Mayoría del partido A en D2



← **GERRYMANDER**

The background is a solid dark red color. In the top-left corner, there are three vertical bars of varying heights, each composed of three overlapping circles. In the bottom-right corner, there are four vertical bars of varying heights, each composed of four overlapping circles. All circles are a lighter shade of red than the background.

Idea fundamental del algoritmo

La idea fundamental es...

... decidir si un determinado distrito debe ser asignado a D1 o a D2.



Imaginemos queda un sólo precinto sin asignar, el precinto P_n

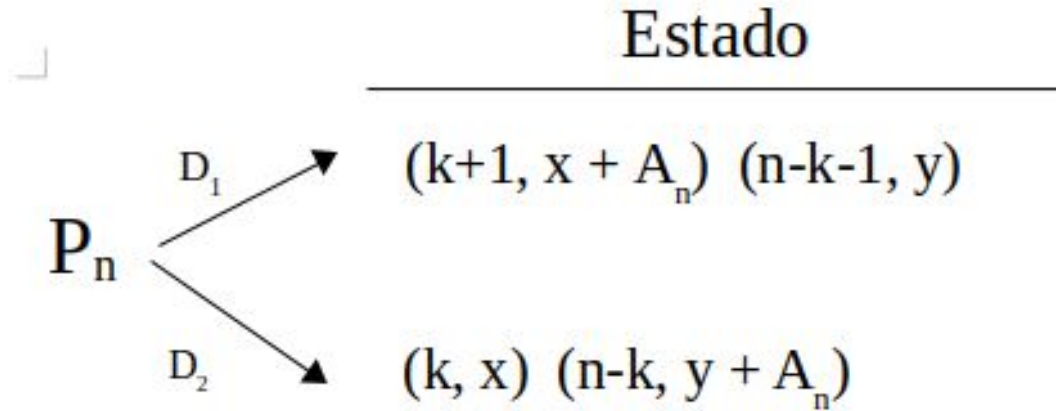
En este caso el estado de los precintos es el siguiente:

- **D1:** k precintos asignados, x votos al partido A.
- **D2:** $n-k-1$ precintos asignados, y votos al partido A.

Estado : $(k, x) (n-k-1, y)$

2 POSIBILIDADES

- Sabemos que el precinto P_n tiene A_n votos al partido A.



NUEVO ESTADO

$$S_{j, k, x, y}$$

j : número de precintos asignados

k : número de precintos asignados a D1

x : número de votos al partido A en D1

y : número de votos al partido A en D2

NUEVO ESTADO

j : número de precintos asignados

k : número de precintos asignados a D1

x : número de votos al partido A en D1

y : número de votos al partido A en D2

S **j, k, x, y = true** sii existe una asignación de los primeros **j precintos** /

- **|D1| = k**
- **A(D1) = x**
- **A(D2) = y**

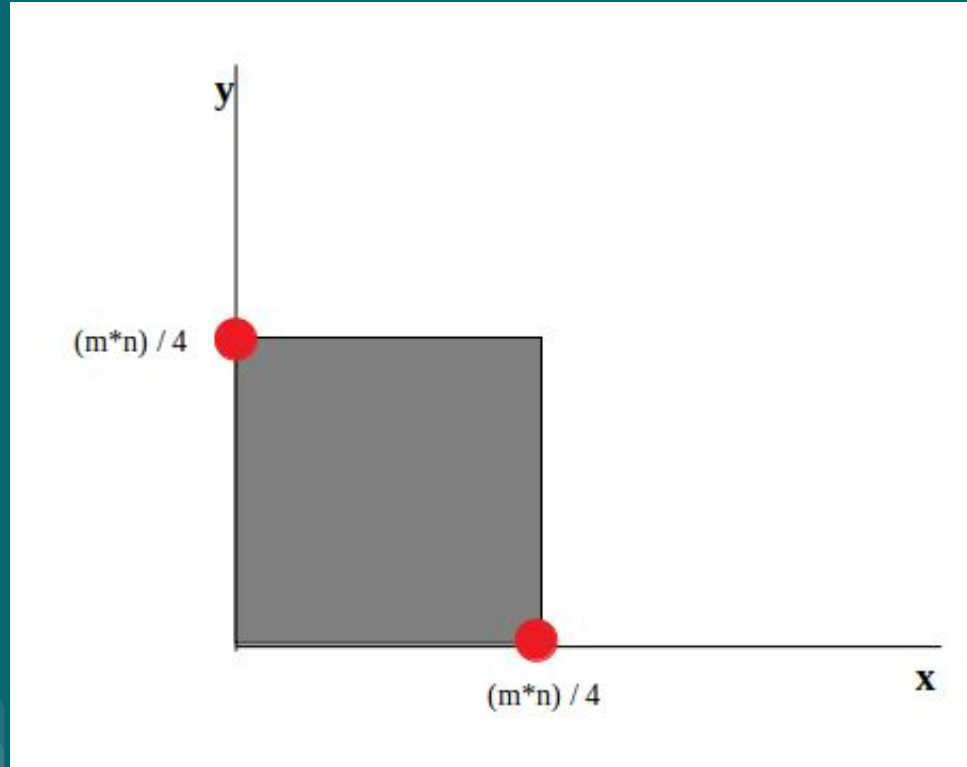
y **false** en caso contrario.

NUEVO ESTADO

$S_{j, k, x, y} = \text{true}$ sii existe una asignación de los primeros j precintos /

- $|D1| = k$
- $A(D1) = x$
- $A(D2) = y$

y **false** en caso contrario.





Un ejemplo

Los siguientes estados están fuera de la zona gris, pero

$S_{4,2,95,90} = \text{true}$

$S_{4,2,94,90} = \text{false}$

| | |
|------------|------------|
| $A_1 = 45$ | $A_2 = 50$ |
| $A_3 = 34$ | $A_4 = 56$ |



Programación dinámica

SUBESTRUCTURA ÓPTIMA

$S_{j, k, x, y} = \text{true}$ sii

A. $S_{j-1, k, x-A_n, y} = \text{true}$ y j se ha asignado a **D1** $\rightarrow c[j][k][x][y] = 1$

B. $S_{j-1, k-1, x, y-A_n} = \text{true}$ y j se ha asignado a **D2** $\rightarrow c[j][k][x][y] = 2$

$$S_{j, k, x, y} = S_{j-1, k, x-A_n} \vee S_{j-1, k-1, x, y-A_n}$$

Pseudocódigo

funcion **gerrymander** (n, m, A₁, A₂, ..., A_n, ady[1..n][1..n]) : gerrymander

{ Crear la tabla de estados **s[n][n/2][m*n][m*n]** y la tabla de resultados intermedios **c[n][n/2][m*n][m*n]** }

mayoría ← (m*n) / 4 + 1

for j ← 1 **to** n

for k ← 0 **to** n/2

for x ← 0 **to** m*n

for y ← 0 **to** m*n

if j = 0

if (x == A_j **and** k = 1 **and** y = 0)

 s[j][k][x][y] ← true

 c[j][k][x][y] ← 1

else if (y == A_j **and** k = 0 **and** x = 0)

 s[j][k][x][y] ← true

 c[j][k][x][y] ← 2

else

 s[j][k][x][y] ← false

else

if (x >= A_j **and** k >= 1 **and** s[j-1][k-1][x-A_j][y])

 s[j][k][x][y] ← true

 c[j][k][x][y] ← 1

else if (y >= A_j **and** s[j-1][k][x][y-A_j])

 s[j][k][x][y] ← true

 c[j][k][x][y] ← 2

else

 s[j][k][x][y] ← false

for x ← mayoría **to** m*n

for y ← mayoría **to** m*n

if s[n][n/2][x][y]

 gerrymander ← **getGerrymander2Distritos** (n, A₁, A₂, ..., A_n, c, x, y)

return gerrymander

return null

Función principal

funcion **gerrymander** (n, m, A_1, A_2, \dots, A_m , $ady[1..n][1..n]$) : gerrymander

{ Crear la tabla de estados $s[n][n/2][m*n][m*n]$ y la tabla de resultados intermedios $c[n][n/2][m*n][m*n]$ }

$\Theta(n^4 m^2)$

mayoria $\leftarrow (m*n) / 4 + 1$

$\Theta(cte)$

for j $\leftarrow 1$ **to** n

for k $\leftarrow 0$ **to** n/2

for x $\leftarrow 0$ **to** m*n

for y $\leftarrow 0$ **to** m*n

if j = 0

if (x == A_j **and** k = 1 **and** y = 0)

$s[j][k][x][y] \leftarrow \text{true}$

$c[j][k][x][y] \leftarrow 1$

else if (y == A_j **and** k = 0 **and** x = 0)

$s[j][k][x][y] \leftarrow \text{true}$

$c[j][k][x][y] \leftarrow 2$

else

$s[j][k][x][y] \leftarrow \text{false}$

else

if (x >= A_j **and** k >= 1 **and** $s[j-1][k-1][x-A_j][y]$)

$s[j][k][x][y] \leftarrow \text{true}$

$c[j][k][x][y] \leftarrow 1$

else if (y >= A_j **and** $s[j-1][k][x][y-A_j]$)

$s[j][k][x][y] \leftarrow \text{true}$

$c[j][k][x][y] \leftarrow 2$

else

$s[j][k][x][y] \leftarrow \text{false}$

$\Theta(cte)$

$\Theta(mn)$

$\Theta(n^2 m^2)$

$\Theta(n^3 m^2)$

$\Theta(n^4 m^2)$

for x \leftarrow mayoría **to** m*n

for y \leftarrow mayoría **to** m*n

if s[n][n/2][x][y]

gerrymander \leftarrow **getGerrymander2Distritos** (n, A₁, A₂, ..., A_n, c, x, y) $\Theta(n)$

return gerrymander

$\Theta(n)$

$\Theta(n)$

$\Theta(n)$

return null

$O(n^4 m^2)$

Obtención de la solución a partir de la tabla de resultados intermedios

```
function getGerrymander2Distritos (c[1.. n][0..k][0..m*n][0..m*n],  
                                     A1,A2, ..., An, xFinal, yFinal) : gerrymander  
  
    j ← n -1  
    k ← n/2  
    x ← xFinal  
    y ← yFinal  
    d1 ← ∅  
    d2 ← ∅  
  
    while j > 0 do  
        if c[j][k][x][y] = 1  
            d1 ← d1 ∪ { j }  
            k ← k -1  
            x ← x - Aj  
        else  
            d2 ← d2 ∪ { j }  
            y ← y - Aj  
  
    return [d1, d2]
```

function **getGerrymander2Distritos** ($c[1..n][0..k][0..m*n][0..m*n]$, A_1, A_2, \dots, A_n , x_{Final} , y_{Final}) : gerrymander

$j \leftarrow n-1$
 $k \leftarrow n/2$
 $x \leftarrow x_{\text{Final}}$
 $y \leftarrow y_{\text{Final}}$
 $d1 \leftarrow \emptyset$
 $d2 \leftarrow \emptyset$

$\Theta(\text{cte})$

while $j > 0$ **do**


if $c[j][k][x][y] = 1$
 $d1 \leftarrow d1 \cup \{j\}$
 $k \leftarrow k-1$
 $x \leftarrow x - A_j$
else
 $d2 \leftarrow d2 \cup \{j\}$
 $y \leftarrow y - A_j$

$\Theta(\text{cte})$

$\Theta(n)$

return $[d1, d2]$

Verificación de la contigüidad de los precintos en cada distrito



```
funcion gerrymanderValido ( d1[1.. n/2], d2[1 .. n/2], ady[1..n, 1..n]) : boolean
    adyD1  $\leftarrow \emptyset$ 
    adyD2  $\leftarrow \emptyset$ 

    for each i in d1
        for j  $\leftarrow$  1 to n
            if ady[i][j] = 1 then adyD1  $\leftarrow$  adyD1  $\cup$  { j }

    for each i in d2
        for j  $\leftarrow$  1 to n
            if ady[i][j] = 1 then adyD2  $\leftarrow$  adyD2  $\cup$  { j }

    if d1  $\subset$  adyD1 and d2  $\subset$  adyD2
        return true
    else
        return false
```

funcion **gerrymanderValido** (d1[1.. n/2], d2[1 .. n/2], ady[1..n, 1..n]) : boolean

adyD1 $\leftarrow \emptyset$
adyD2 $\leftarrow \emptyset$ $\Theta(\text{cte})$

for each i in d1

for j \leftarrow 1 to n

if ady[i][j] = 1 **then** adyD1 \leftarrow adyD1 U { j } $\Theta(\log n)$ $\Theta(n \cdot \log n)$

$\Theta(n^2 \log n)$

for each i in d2

for j \leftarrow 1 to n

if ady[i][j] = 1 **then** adyD1 \leftarrow adyD1 U { j } $\Theta(\log n)$ $\Theta(n \cdot \log n)$

$\Theta(n^2 \log n)$

if d1 \subseteq adyD1 **and** d2 \subseteq adyD2
 return true
else
 return false

$\Theta(\log n)$

Código





¿Preguntas?