

## MagCuadrados

2. Un cuadrado mágico de orden  $n$ , es un cuadrado con los números  $\{1, \dots, n^2\}$ , tal que todas sus filas, columnas y las dos diagonales suman lo mismo (ver figura). El número que suma cada fila es llamado número mágico.

2	7	6
9	5	1
4	3	8

Existen muchos métodos para generar cuadrados mágicos. El objetivo de este ejercicio es contar cuántos cuadrados mágicos de orden  $n$  existen.

- a) ¿Cuántos cuadrados habría que generar para encontrar todos los cuadrados mágicos si se utiliza una solución de fuerza bruta?

### Ejercicio de conteo

② Hacía que generas  $(n^2)$  cuadrados. Esto es ya que para la primera posición tienen  $n^2$  posibilidades para la segunda  $n^2 - 1$ , y así hasta la última donde es 1 de números distintos (no se pueden repetir)

- b) Enunciar un algoritmo que use *backtracking* para resolver este problema que se base en la siguientes ideas:

- La solución parcial tiene los valores de las primeras  $i - 1$  filas establecidos, al igual que los valores de las primeras  $j$  columnas de la fila  $i$ .
- Para establecer el valor de la posición  $(i, j+1)$  (o  $(i+1, 1)$  si  $j = n$  e  $i \neq n$ ) se consideran todos los valores que aún no se encuentran en el cuadrado. Para cada valor posible, se establece dicho valor en la posición y se cuentan todos los cuadrados mágicos con esta nueva solución parcial.

Mostrar los primeros dos niveles del árbol de *backtracking* para  $n = 3$ .

- Cuáles son las soluciones posibles? Todos los cuadrados de  $n \times n$  que tengan números distintos en sus casillas
- Cuáles son las soluciones lanchadoras? Aquellos cuadrados que sean mágicos
- Cómo se construyen esas soluciones? Fijarse el 1º número y luego cuál podría ser el segundo y así

- 1º intento a generar TODOS los cuadrados, después pedir

Solución parcial

→ 10, 11 devuelve 0, si NO es 1 se

cb: esMagico(c)

$$(l, j) = (N+1, 1)$$

$CM(c, l, j, N) \left\{ \sum_{n \in \text{NumerosValidos}(r)} CM(c[l:j], n, \text{sigPos}[l:j], N) \right.$

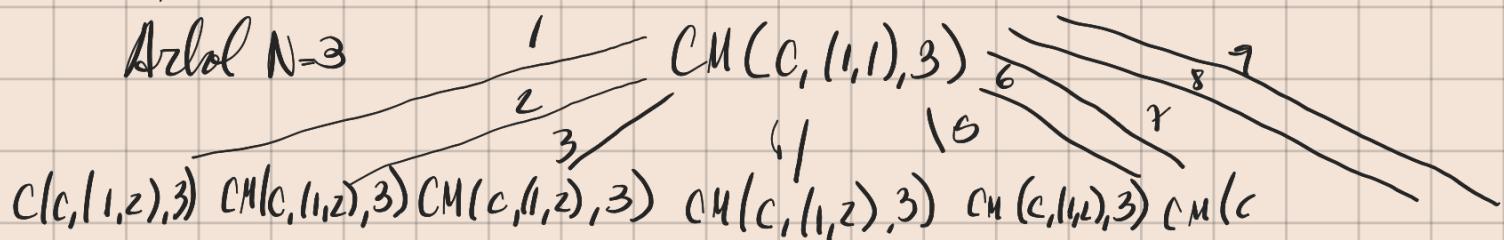
↓ devuelven los  
números que  
NO pisan

↓ modifica la pos sig  
orden

cc

La llamamos  $R \leftarrow CM(c, l, j)$

Árbol N=3



↪ cada nodo tiene 8 ramas

CuadradoMágico(C,i,j,N)

CantidadDeMágicos = 0

If i = N+1 then

If esMágico(C) then

    return 1

else

    return 0

endif

else

    for k in numsDisponibles(C)

        If j != N then

            cantidadDeMágicos += CuadradoMágico(C[i,j] <-  
                k, i, j+1, N)

        else

            cantidadDeMágicos += CuadradoMágico(C[i,j] <-  
                k, i+1, 0, N)

        endif

    endfor

    return cantidadDeMágicos

endif

- c) Demostrar que el árbol de *backtracking* tiene  $\mathcal{O}((n^2)!)$  nodos en peor caso.

En el peor caso tendría que querer todos los nodos. Como tengo  $n^2$  lugares para llenar, un árbol va a tener una altura igual a  $n^2$  ya que cada nivel corresponde a una posición a llenar. Para la primera posición voy a tener  $n^2$  números posibles, por lo tanto, los nodos del nivel 1 (nodo) están la función empieza a gigantearse) van a tener  $n^2$  hijos. Los del nivel 2 (como ya se usó un número) van a tener  $n^2-1$  hijos y así...

$$\rightarrow \# \text{ nodos} = n^2(n^2-1)(n^2-2) \dots 1 = (n^2)!$$

Mis  $n^2$  nodos tienen  
 $n^2-1$  hijos y  
así

- d) Considere la siguiente poda al árbol de *backtracking*: al momento de elegir el valor de una nueva posición, verificar que la suma parcial de la fila no supere el número mágico. Verificar también que la suma parcial de los valores de las columnas no supere el número mágico. Introducir estas podas al algoritmo e implementarlo en la computadora. ¿Puede mejorar estas podas?

Podas: al momento de elegir el valor de una posición verificar que la suma parcial de la fila no supere al número mágico, verificar tambien lo mismo para las columnas

Podas: chequear lo mismo para las diagonales, (lo de nums disponibles ya es una poda también ya que explora ciertos números selectos), que las filas y columnas vayan sumando lo mismo

CuadradoMágico(C,i,j,N, sumParDiag)

CantidadDeMágicos = 0

If i = N+1 then

If esMágico(C) then

    return 1

else

    return 0

endif

else

    for k in numsDisponibles(C)

        if sum(C[i, :j]) + k <= numMagico(N) && sum(C[:i, j])  
        + k <= numMagico(N) && sumParDiag + k <=  
        numMagico(N)

            if esPosDiagonal (i,j) then

                sumParDiag += k

            endif

            If j != N then

                cantidadDeMágicos += CuadradoMágico(C[i,j]  
                <- k, i, j+1, N, sumParDiag)

            else

                cantidadDeMágicos += CuadradoMágico(C[i,j]  
                <- k, i+1, 0, N, sumParDiag)

            endif

            endif

    endfor

    return cantidadDeMágicos

endif

Preg ¿¿¿Cuando dice que no supere el número mágico se refiere a que la suma parcial no sea mayor a la de otra fila ya completa o al número mágico del cuadrado per se???

A la fila ya completa, cuando completamos la 1<sup>ra</sup> fila es lo a ser un cuadrado a número mágico

```
CuadradoMágico(C,i,j,N, sumParDiag, numMagicoParcial)
CantidadDeMágicos = 0
If i = N+1 then
    If esMágico(C) then
        return 1
    else
        return 0
    endif
else if i = 2
    numMagico = sumaFila1
Else
    for k in numsDisponibles(C)
        if sum(C[i, :j]) + k <= numMagico && sum(C[:i, j]) + k
            <= numMagico && sumParDiag + k <= numMagico
            if esPosDiagonal (i,j) then
                sumParDiag += k
            endif
            If j != N then
                cantidadDeMágicos += CuadradoMágico(C[i,j]
                    <- k, i, j+1, N, sumParDiag, numMagico)
            else
                cantidadDeMágicos += CuadradoMágico(C[i,j]
                    <- k, i+1, 0, N, sumParDiag, numMagico)
            endif
        endif
    endfor
    return cantidadDeMágicos
endif
```

- e) Demostrar que el número mágico de un cuadrado mágico de orden  $n$  es siempre  $(n^3 + n)/2$ . Adaptar la poda del algoritmo del ítem anterior para que tenga en cuenta esta nueva información. Modificar la implementación y comparar los tiempos obtenidos para calcular la cantidad de cuadrados mágicos.

¿Demo con inducción?  $\hookrightarrow$  álgebra

CuadradoMágico(C,i,j,N,numMagico)

    numMagico2 = numMagico

    CantidadDeMágicos = 0

// el caso base anterior no hace falta ya que los cuadrados mágicos son cuadrados donde todas sus filas suman el número y quizas se permutan entre sí

If j = N && numMagico != 0 then

// si la fila no suma el número mágico exacto. chau.

Mejoré la poda de arriba que solo pedía menor o igual

    return 0

elif i = N+1

    return 1

else

    for k in numsDisponibles(C)

        if numMagico - k >= 0 // si la suma parcial ya es mayor que el número mágico no pongo ese número

        If j != N then

            cantidadDeMágicos += CuadradoMágico(C[i,j]  
            <- k, i, j+1, N, numMagico - k)

        else

            cantidadDeMágicos += CuadradoMágico(C[i,j]  
            <- k, i+1, 0, N, numMagico2-k)

```
    endif  
endfor  
return cantidadDeMágicos  
endif
```

