A y C

tiempo ejecución

ej:

```
for i = 1 to n do
  for j = 1 to n do
    A[i, j] = 0
```

$T(n) = \sum_{i=1}^{n} \left(\sum_{j=1}^{n} (1) \right) = \sum_{i=1}^{n} n = n^2$

Orden : $O(n^2)$

ej:

$T(n) = \sum_{j=1}^{n} (1+1) = 2n \qquad O(n)$

- si juntamos los 2 ejemplos anteriores

$T(n) = T_1(n) + T_2(n) = n^2 + 2n$

$O = max(O_1, O_2) = O(n^2)$

ej:

```
if A[1, 1] = 0 then
  for i = 1 to n do
    for j = 1 to n do
      A[i, j] = 0
else
  for i = 1 to n do
    A[i, i] = 1
```

$T(n) = 1 + max(n^2, n) = 1 + n^2 \qquad O(n^2)$

ej:

$T(n) = \sum_{i=1}^{n} \left(1 + \sum_{j=1}^{n} (1+1) + 1 + 1 + 1\right) = \sum_{i=1}^{n} \left(4 + \sum_{j=1}^{n} (2) \right) = \sum_{i=1}^{n} (4 + 2n) =$

$= (4 + 2n) \cdot n = 4n + 2n^2 \qquad O(n^2)$

ejm Recursividad

Proc Ejercicio(n: entero, V: tipovector)
Si (n>0) y (n ≤ dim) entonces
Desde i=1 hasta (n-1) hacer
V[i] = V[i+1]
fdesde
Ejercicio(n-1, V)
fsi
fproc

- $T(n) = 1 + n + T(n-1)$

$T(n) = x^n$

$x^n = 1 + n + x^{n-1}$

$x^n - x^{n-1} = 1 + n$

(1) Resuelvo homogenea

$x^n - x^{n-1} = 0$

- Saco factor común

$x^{n-1}(x-1) = 0$

- Raices

$x = 1 \Rightarrow x^{(h)} = A \cdot \cancel{1^n} = A$

(2) Resuelvo particular

$x^n - x^{n-1} = (1+n)$

$x^n - x^{n-1} = (1+n) \cdot 1^n$ Las raices son las bases de las exponenciales

$x = 1$

$x^{(p)} = (B + Cn) \cdot 1^n$

(*) Si raiz de particular es = a una de las homogeneas

⇓

Multiplico por n

$x^{(p)} = (Bn + Cn^2) \cdot \cancel{1^n} = Bn + Cn^2$

$x = x^{(h)} + x^{(p)} = A + Bn + Cn^2 \Rightarrow O(n^2)$

Hoja problemas

1/b) $T(n) = 4n + 3 + T(n-1) + 2T(n-2)$

$T(n) - T(n-1) - 2T(n-2) = 4n + 3$

$T(n) = x^n$

$x^n - x^{n-1} - 2x^{n-2} = 4n + 3$

① homogenea

$x^n - x^{n-1} - 2x^{n-2} = 0$

Saco factor común

$x^{n-2}(x^2 - x - 2) = 0$

$x^2 - x - 2 = 0 \qquad x = \frac{1 \pm \sqrt{1^2 - 4 \cdot 1 \cdot -2}}{2} = \frac{1 \pm 3}{2}$ → $x = 2$, → $x = -1$

$x^{(h)} = A \cdot 2^n + B(-1)^n$

② Particular

$x^n - x^{n-1} - 2x^{n-2} = 4n + 3$ imagina $(4n+3) \cdot 1^n$

Raíces

$x = 1$

$x^{(p)} = (Cn + D) \cdot 1^n = Cn + D$

$x = x^{(h)} + x^{(p)} = A \cdot 2^n + B(-1)^n + Cn + D \Rightarrow O(2^n)$

1/D) $T(n) = T(n/2) + n^2 + 2n$

$T(n) - T(n/2) = n^2 + 2n$

$n = 2^k$

$T(2^k) - T(2^k/2) = (2^k)^2 + 2 \cdot 2^k$

$x^k = T(2^k)$

$x^k - x^{k-1} = 4^k + 2 \cdot 2^k$

① homogenea

$x^k - x^{k-1} = 0$

Factor común

$x^{k-1}(x-1) = 0$

$x = 1$

$x^{(h)} = A$

② Particular

$x^k - x^{k-1} = 4^k + 2 \cdot 2^k$

Raíces

$x = 2$

$x = 4$

$x^{(P)} = B \cdot 4^k + C \cdot 2^k$

$x = x^{(h)} + x^{(P)} = A + B \cdot 4^k + C \cdot 2^k$

$x = A + B \cdot n^2 + C \cdot n \quad \Rightarrow O(n^2)$

2)
$$t(n) = 1+1 + t(n/2) + t(n/2) + 1 + 1 + \sum_{i=1}^{n} (1 + \max(1+1, 1+1)) + \sum_{j=1}^{n} (1)$$

$$t(n) = 4 + 4n + 2t(n/2)$$

$$t(n) - 2t(n/2) = 4 + 4n$$

$$n = 2^k$$

$$T(2^k) - 2t\left(\frac{2^k}{2}\right) = 4 + 4 \cdot 2^k$$

$$T(2^k) - 2t(2^{k-1}) = 4 + 4 \cdot 2^k$$

$$x^k = t(2^k)$$

$$x^k - 2x^{k-1} = 4 + 4 \cdot 2^k$$

① homogenea

$$x^k - 2x^{k-1} = 0 \Rightarrow x^{k-1}(x-2) = 0$$

Raiz:

$$x = 2$$

$$x - 2 = 0$$

$$x = 2$$

$$x^{(h)} = A \cdot 2^k$$

② particular

$$x^k - 2x^{k-1} = 4 + 4 \cdot 2^k \quad \rightarrow \cdot 1^k$$

$$x = 1$$

$$x = 2$$

$$x^{(p)} = B + C \cdot 2^k \cdot k$$

$$x = A \cdot 2^k + B + C \cdot 2^k \cdot k$$

$$x = A \cdot n + B + C \cdot n \cdot \log_2 n \Rightarrow O(n \cdot \log_2 n)$$

Tema-2-

- Algoritmos Voraces
  - Entrada de N candidatos
  - Un subconjunto de estos satisface las restricciones
  - obtener solución que maximice o minimice la función

- En cada iteración solo puedo ver los datos y trabajar con los de esa iteración (Ni anteriores ni posteriores)

ej:

- Problema de las monedas
- Problema de la mochila y el ladrón

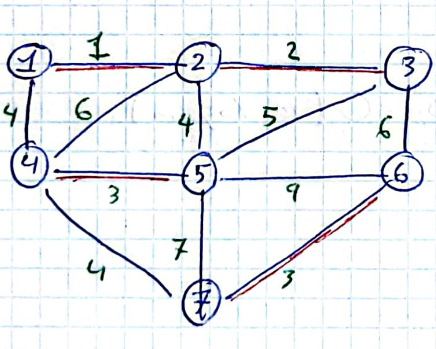
	1	2	3	4	5
Valor	20	30	66	40	60
Peso	10	20	30	40	50
Valor/Peso	2	1'5	2'2	1	1'2

- 3 posibilidades:
  - 1º > peso
  - 1º > valor
  - 1º > $\frac{\text{valor}}{\text{peso}}$

- Ordenar un conjunto de valores

$$O(n \cdot \log n)$$

- Kruskal
  - Aristas de menor peso (sin formar ciclos)



1 {1,2}
2 {2,3}
3 {4,5}
4 {6,7}
5 ''
6 ''
7 ''

- Prim
  - Selecciono un nodo cualquiera y selecciono las aristas de menor peso que esten relacionados con este

Kruskal

func Kruskal ($G=\langle N,A \rangle$: grafo , longitud : $A \to \mathbb{R}^+$) : aristas

Ordenar A por longitudes crecientes
n = Número de nodos que hay en N
$T = \emptyset$

repetir
e = (u,v) {arista mas corta, aún no considerada}

compU = buscar(u)
compV = buscar(V)
si (compU ≠ compV) entonces
fusionar (compU, compV)
$T = T \cup \{e\}$
Hasta que T contenga n-1 aristas
Devolver T

Divide y vencerás

K llamadas recursivas
p = la expresión no recursiva, mayor grado
b = tamaño en que se divide la llamada recursiva

coste $t(n) = k \cdot t\left(\frac{n}{b}\right) + g(n)$

$$\text{si} \begin{cases} k < b^p & O(n^p) \\ k = b^p & O(n^p \cdot \log n) \\ k > b^p & O(n^{\log_b k}) \end{cases}$$

$x^{3} + 2x + 5 \Rightarrow p = 3$

Ej:
1 2^0
$1 = 2^0 \Rightarrow O(n^p \cdot \log n) = O(\log n)$

Merges hoet

- $k=2$ llamadas recursivas
- $p=1$ $\quad n=1$
- $b=2$ (Mitad del problema)

$2=2^1 \Rightarrow O(n^1 \cdot \log n)$

Tema -4- Programación dinámica

- Ejecuto varias secuencias y al final del todo sabré cual es la mejor.

~~1º objeto peso 6U y valor 8 cada uno~~
~~2 " " 5U " " 5 " "~~

Mochila Máx 11 U

lim. peso	0	1	2	3	4	5	6	7	8	9	10	11
$w_1=1$ $V_1=1$	0	1	1	1	1	1	1	1	1	1	1	1
$w_2=2$ $V_2=6$	0	1	6	7	7	7	7	7	7	7	7	7
$w_3=5$ $V_3=18$	0	1	6	7	7	18	19	24	25	25	25	25
$w_4=6$ $V_4=22$	0	1	6	7	7	18	22	23	28	29	29	40
$w_5=7$ $V_5=28$	0	1	6	7	7	18	22	28	29	34	35	40

$\text{Celda} = \max(V[f-1][c],\ V[f-1][c-w_i])$

$V(4,11) \neq V(3,11)$ pero $\underbrace{V(4,11) = V(3,11-w_4)+V_4}$

Una carga óptima debe incluir el objeto 4

$V(3,5) \neq V(2,5)$ pero $\underbrace{V(3,5) = V(2,5-w_3)+V_3}$

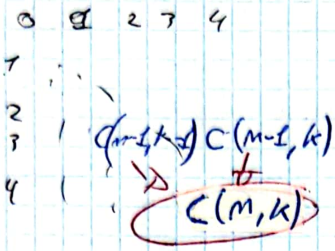
Por lo que lo más óptimo es pillar el 3 y el 4 solo

- Cálculo coeficiente binomial

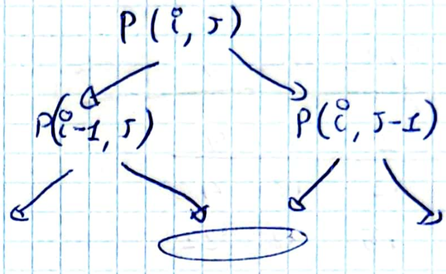
$0 \leq k \leq n$

$$\binom{n}{k} = \begin{cases} 1 & k=0 \text{ ó } k=n \\ \binom{n-1}{k-1} + \binom{n-1}{k} & 0<k<n \\ 0 & \text{Resto} \end{cases}$$

```
fun C(n, k)
  si k == 0 ó k == n entonces
    Devolver 1
  sino
    Devolver C(n-1, k-1) + C(n-1, k)
ffunc
```




```
fun P(i, j)
  si i == 0 entonces
    Devolver 1
  sino
    si j == 0 entonces
      Devolver 0
    sino
      Devolver pP(i-1, j) + qP(i, j-1)
    fsi
  fsi
ffun
```



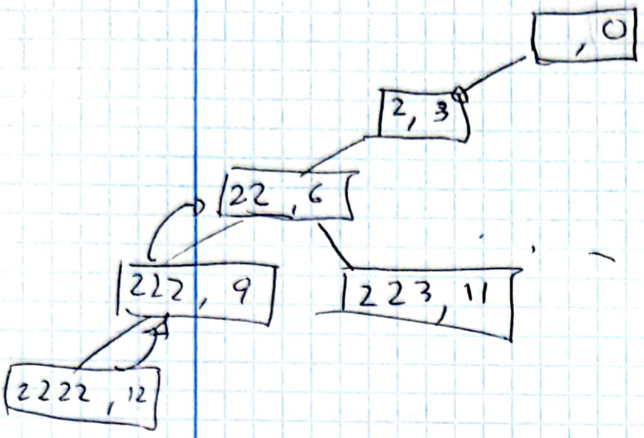
Tema 5 BackTracking

- tablero

```
for fila {
  for columna {
    if (posicionPosible) {
      //Recursiva
    }
  }
}
```

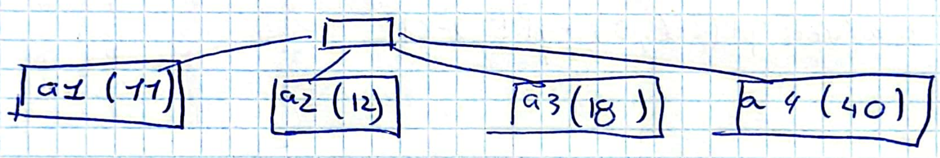
- Reducción de frase   acabada

nochila $\begin{array}{c|cccc} w & 2, & 3, & 4, & 5 \\ v & 3 & 5 & 6 & 10 \end{array}$ $w = 8$



- **Ramificación y poda**

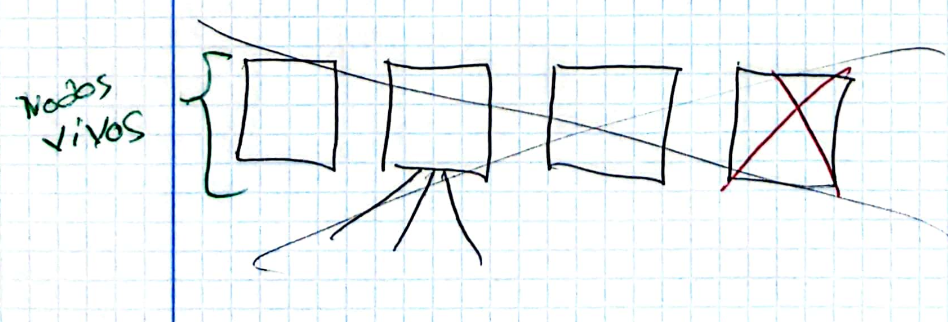
	1	2	3	4
a	11	12	18	40
b	14	15	13	22
c	11	17	19	23
d	17	14	20	28

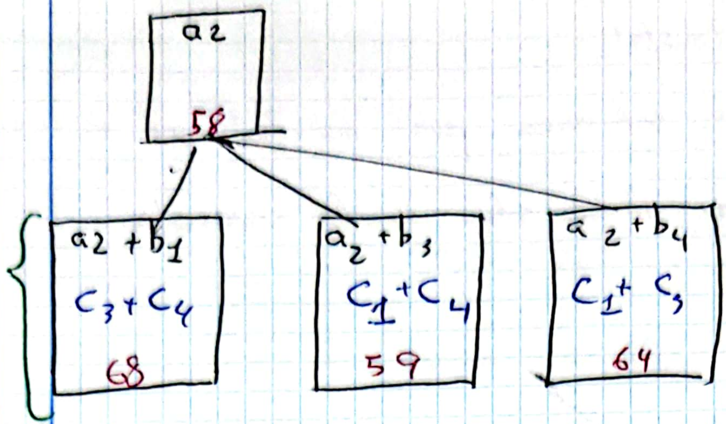


- A partir de estos, buscamos las soluciones de menor peso posibles

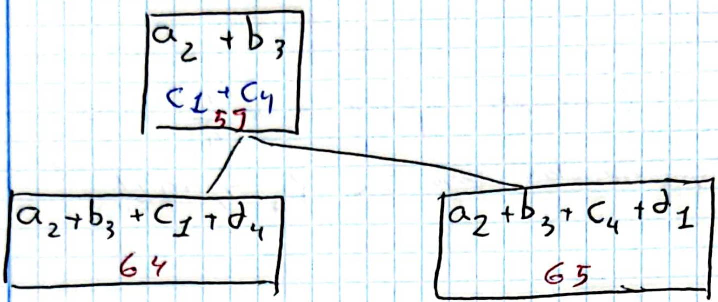
a1 +	a2 +	a3 +	a4 +
$d_2 + b_3 + b_4$	$c_1 + b_3 + b_4$	$c_1 + d_2 + b_4$	$c_1 + d_2 + b_3$
cota 60	58	65	78

- vemos que aunque NO son soluciones completas, vemos que el último como mínimo va a tener 78 → podamos
- De los otros 3, selecciono el mejor y sigo mirando B

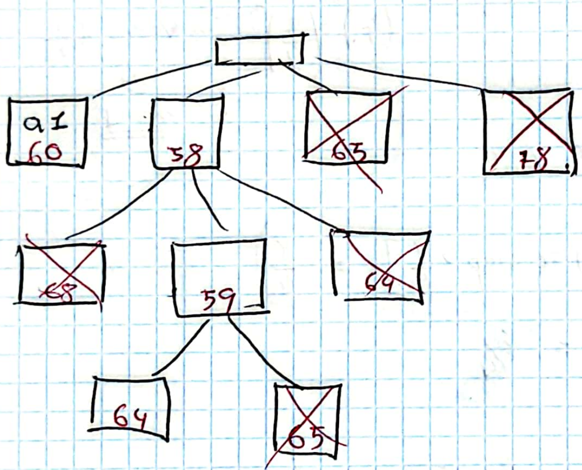




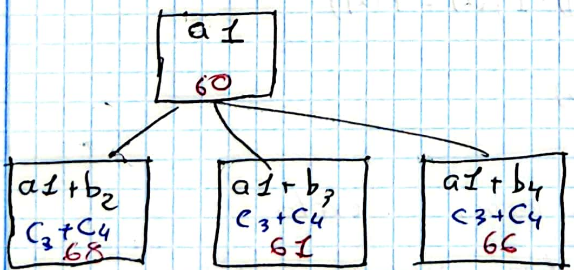
- De los nodos vivos, escojo el de menor cota y sigo



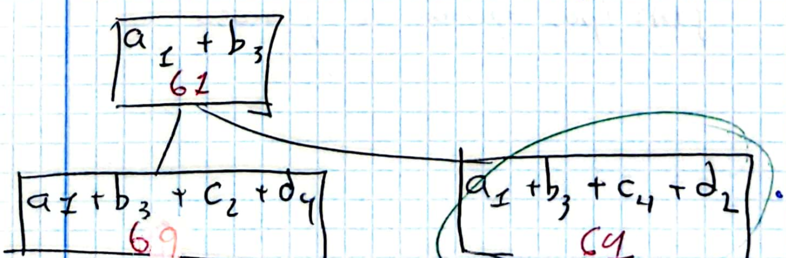
- Al encontrar una solución posible, podo

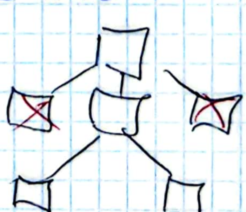


- Queda un nodo vivo que tengo que expandir



- Puedo podar pero sigo expandiendo el nodo vivo





- podo y encuentre el mejor ya que no quedan vivos

Algoritmos NO deterministas

- Montecarlo: siempre da solución pero no siempre es correcta
- Vegas: puede fallar pero si da solu., es la correcta

- Montecarlo:

- Dada una muestra aleatoria y vemos si satisface las necesidades en vez de comprobar todos los elementos.

Ej:

Dado un vector, ver si 1 elemento aparece + de $\frac{1}{4}$ veces

¿cuantas veces hay que ejecutar para que $P_A \geq 90\%$?

2 casos

$$\begin{cases} - \text{No hay ele} > \frac{1}{4}; \quad P_A = 1 \\ - \text{Si hay ele} > \frac{1}{4} \end{cases}$$

$\Downarrow$

tomando muestra aleatoria

$$\begin{cases} - P_A = [P(\text{muestra}) \text{ se rep} > \frac{1}{4}] = \frac{1}{4} \\ - P_F = -P_A + 1 = 3/4 \end{cases}$$

para que $P_A > \frac{9}{10} \Rightarrow P_F = \frac{1}{10}$

$(P_F)^n = \left(\frac{3}{4}\right)^n$ — n veces $\leq \frac{1}{10}$

$\Downarrow$

$\underline{\underline{n = 9}}$

- Vegas

- Se ejecuta hasta que de una solución correcta

$$F.\ Vegas = p(x) \cdot s(x) + (1 - p(x)) \cdot (f(x) + t(x))$$

$$F.\ Vegas = s(x) + \frac{1 - p(x)}{p(x)} \cdot f(x)$$

$\Downarrow$

$t(x)$

$P(x)$ = proba. de éxito

$s(x)$ = tpo. esperado de éxito

$t(x)$ = tpo. esperado total

$f(x)$ = tpo. esperado de fracaso

Ej:

• colón descubre américa

$P(x) = 80\%$

$S(x) = 10s$

$t(x) = 60s$

$$F.vegas = S(x) + \frac{1-P(x)}{P(x)} \cdot f(x)$$

$$F.vegas = \left(10 + \frac{1-0'8}{0'8}\right) \cdot f(x)$$

$$f(x) = \frac{60}{\left(10 + \frac{1-0'8}{0'8}\right)} = \frac{60}{10'25} = 5'85$$