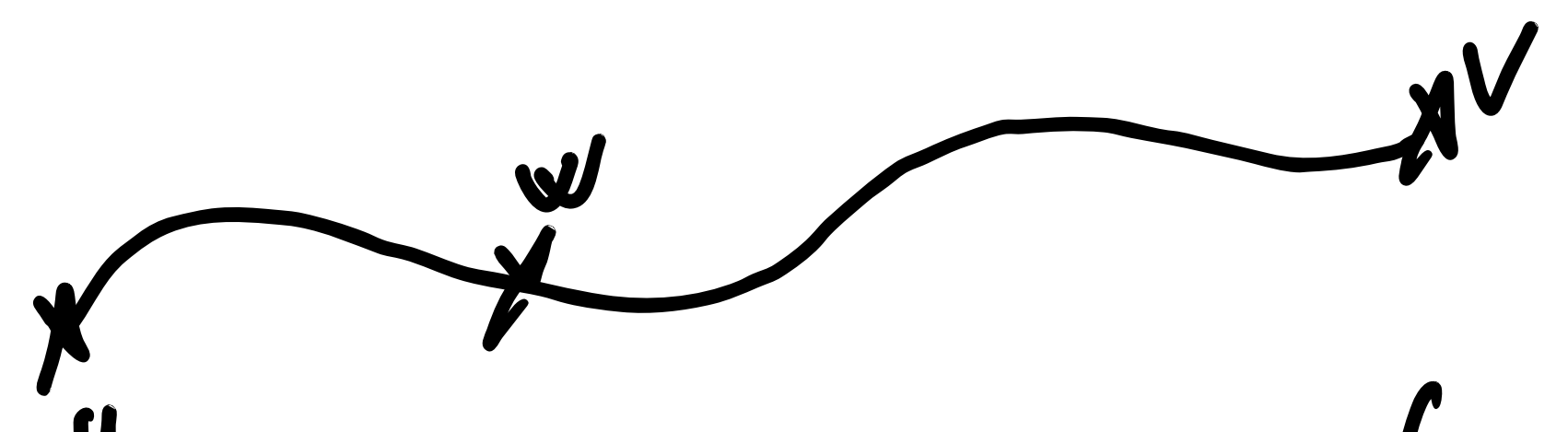


// Pour avoir bonne solt^o, nécessaire d'avoir des sous-solt^o optimaux

ex: PCE (plus court chemin) 

// Rq: PLC \rightarrow pas possible utiliser prog dynamique (car pas m propriétés)

Algo prog dynamique efficace?

non ex: subsetsum $O(nS)$ // coût pas mieux pas 3 sur m

Prob Rendue de monnaie

Algo Glanton OK \neq mais ~~autre~~ ^{Stirling} avant 71

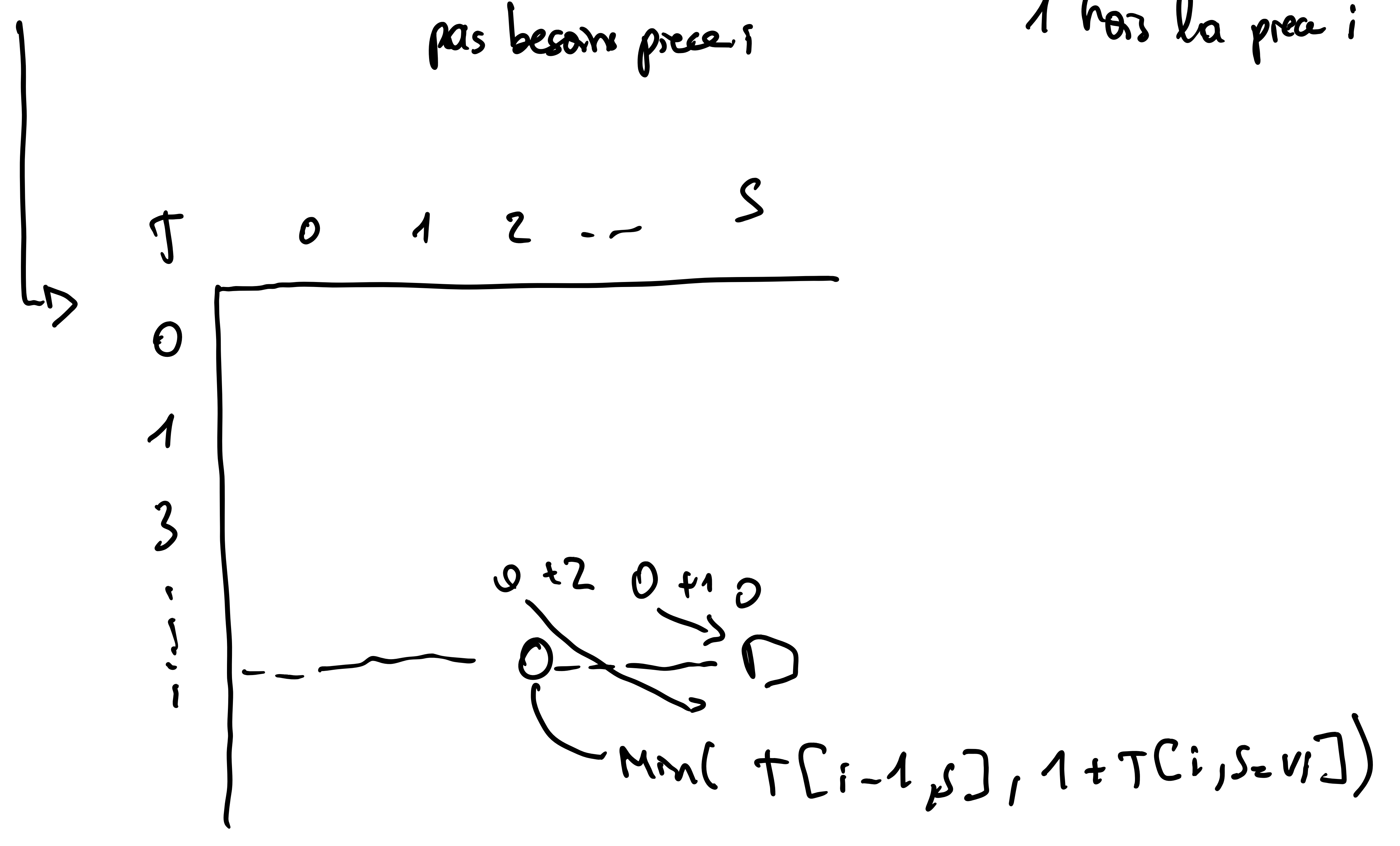
back-tracking pr résoudre prob livre sterling avant 71.

$T[n, s]$ = nb min de pièces de $\{v_1 \dots v_n\}$ pour faire la somme
cas de base:

- si 0 \rightarrow return 0
- si val impossible \rightarrow représentable par -1 / null / ∞ ...

$T[i, s] = ?$ nb min de pièces de $\{v_1 \dots v_i\}$ pour faire s

$$T[i, s] = \text{Min} \left(\underset{\substack{\uparrow \\ \text{pas besoin pièce } i}}{T[i-1, s]}, \quad \underset{\substack{\uparrow \\ \text{1 fois la pièce } i}}{1 + T[i-1, s-v_i]}, \quad \underset{\substack{\uparrow \\ \text{2 fois}}}{2 + T[i-1, s-2v_i]}, \dots \right)$$



Recap:

$T[i,s] = \min(T[i-1,s], 1 + T[i,s-v_i])$

$T[i,0] = 0$

$T[0,s] = 1 - 1/\infty$

Non utilisé car n'apparaît pas en utilisant min.

ex: $T[1,4,5,10]$ $S = 13$

	i s	0	1	2	3	4	5	6	7	8	9	10	11	12	13		
		0	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	Qu'est le + petit? ↳ le prendre + 1	
VB	1	1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	Peut faire 4 avec pièce 1.
	4	2	0	1	2	3	1	2	3	4	2	3	4	5	3	4	Impossible de regarder à droite val 3 ⇒ prendre val du haut
	5	3	0	1	2	3	1	1	2	3	2	2	2	3	3	3	
	10	4	0	1	2	3	1	1	2	3	2	2	2	1	3	3	

pour $v_i \leq s$ $T[1,s] = \min(T[0,s], 1 + T[1,s-1])$ (première ligne)
 $= 2$ $T[2,s] = \min(T[1,s], 1 + T[2,s-4])$ (2ème ligne)

ex cas $(s,i) = (4,2) \Rightarrow \min(T[1,4], 1 + T[2,4-v_2]) = 1$
 $= 3$ $T[3,s] = \min(\dots)$

Rq: pas de changement de $s = 1 \rightarrow 10$ car $10 >$

- On constate 3 en 13, 10
- Comment retrouver le type de pièce utilisé?
- Il faut remarquer pr trv comment a été fait le 3. Noté @

init } Nb = tab de taille $(n+1) \times (s+1)$
 Pour $j = 1$ à s Nb $[0, j] = \infty$
 Pour $i = 0$ à n Nb $[i, 0] = 0$

calcul } Pour $i = 1$ à n
 Pour $j = 1$ à s
 Si $v_i \leq j$ Alors Nb $[i, j] = \min(Nb[i-1, j], 1 + Nb[i, j-v_i])$
 Sinon Nb $[i, j] = Nb[i-1, j]$

$O(Sn)$

Passage tab unidim.

init } Nb = tab de taille $(s+1)$
 Pour $j = 1$ à s : Nb $[j] = \infty$
 Nb $[0] = 0$

calcul } Pour $i = 1$ à n
 Pour $j = 1$ à s
 Si $v_i \leq j$ Alors Nb $[j] = \min(Nb[j], 1 + Nb[j-v_i])$
 Sinon Nb $[j] = Nb[j]$

Prob graph:

// PPC • Bellman Ford (cf cours)

! à 2 poids neg on utilise pas d'algo.

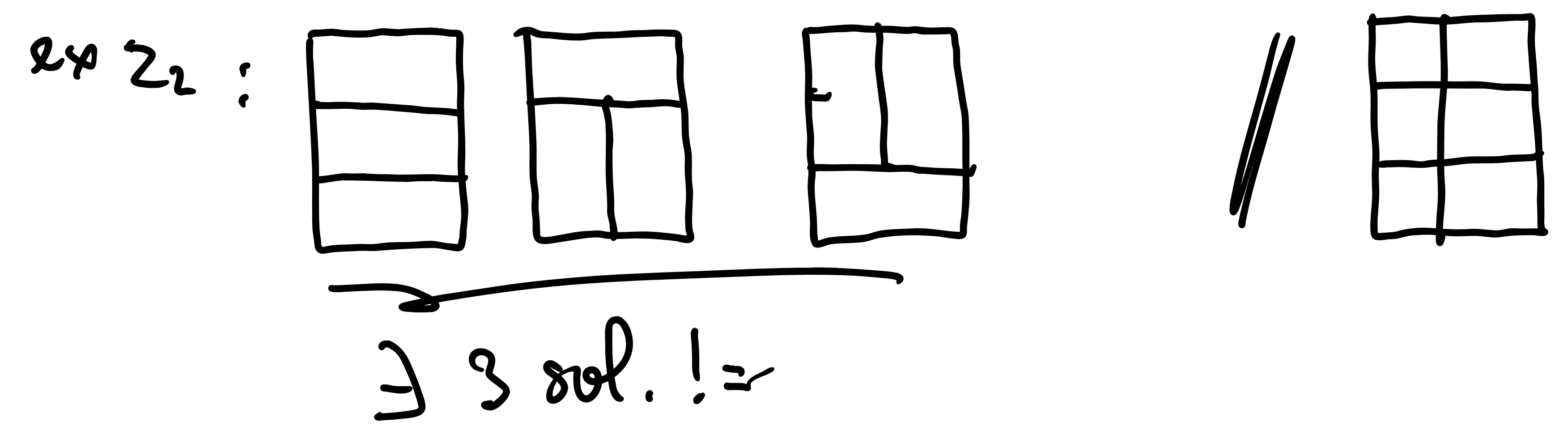
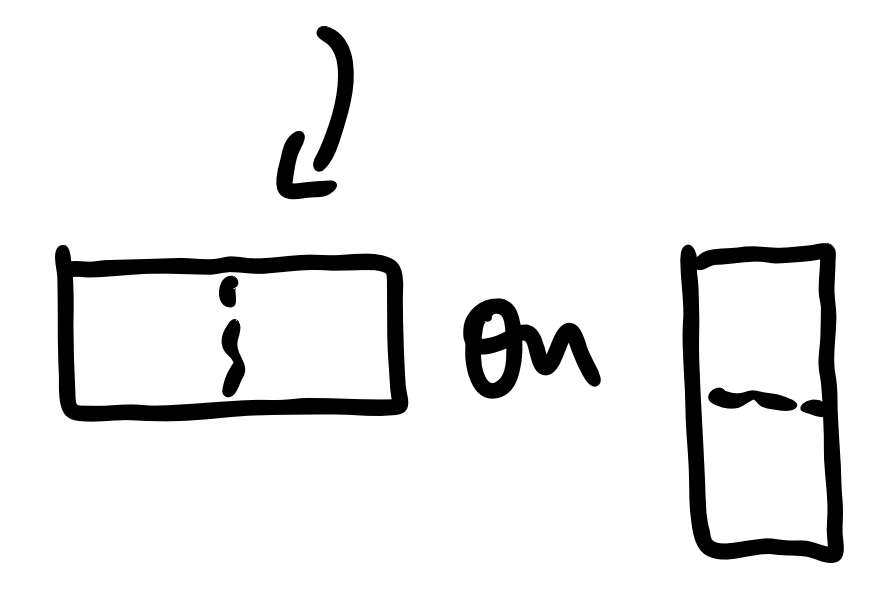
↳ prog dynamique

• Floyd Marshall (cf cours)

↳ prog dynamique

~ Histories de pavages...

On veut compter le nb de pavages distincts d'une zone $3 \times n$ par des dominos 2×1 :



ex : ? → impossible = 0 car il y a un nbr impair de cases.

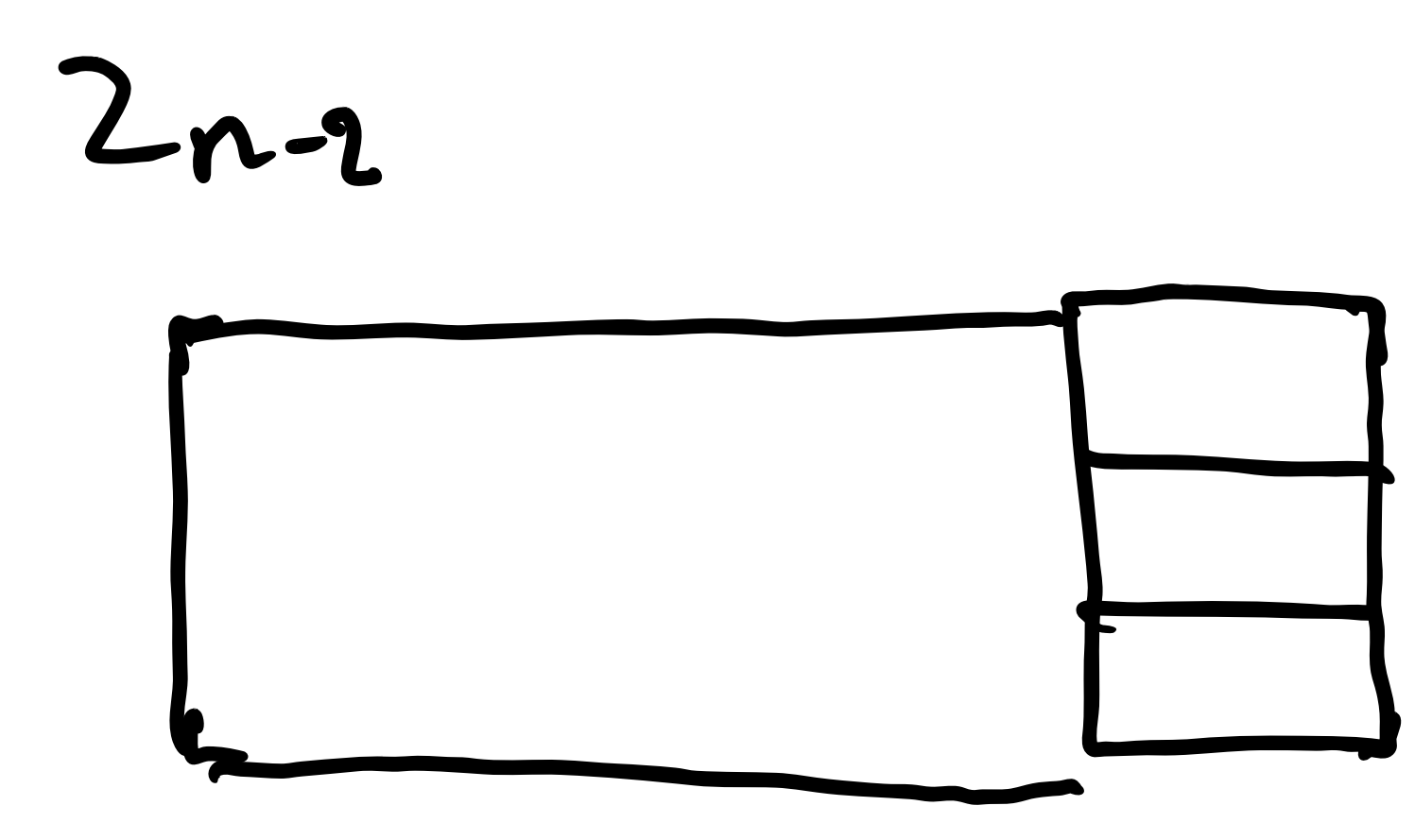
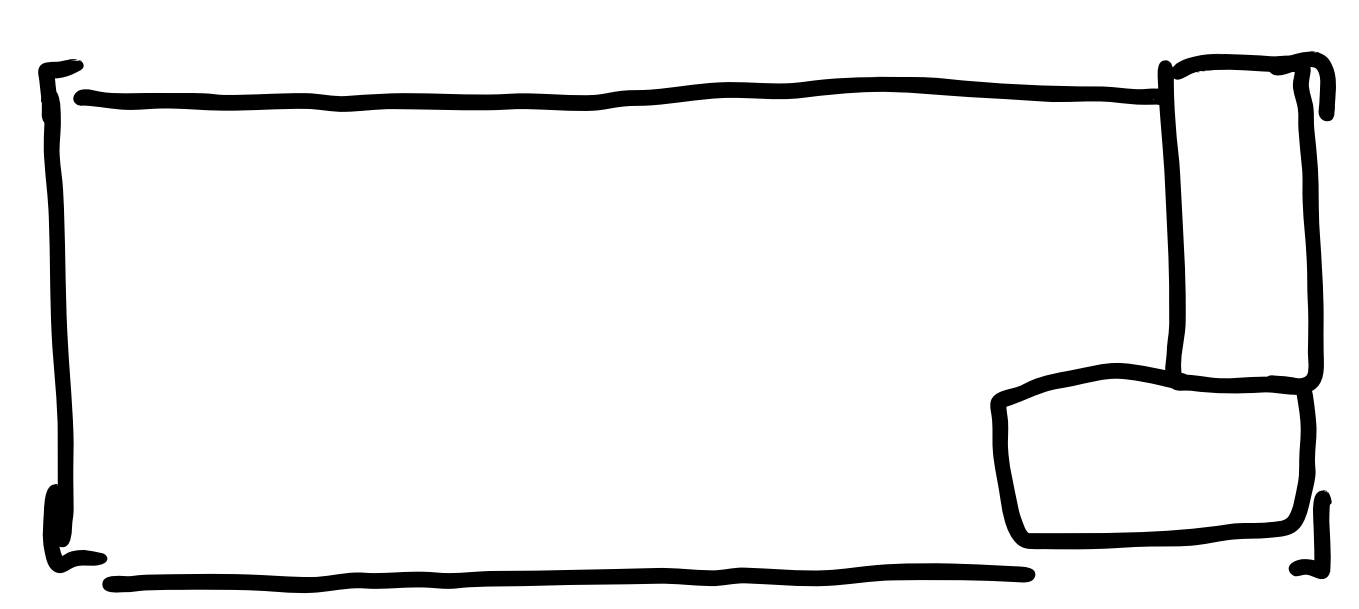
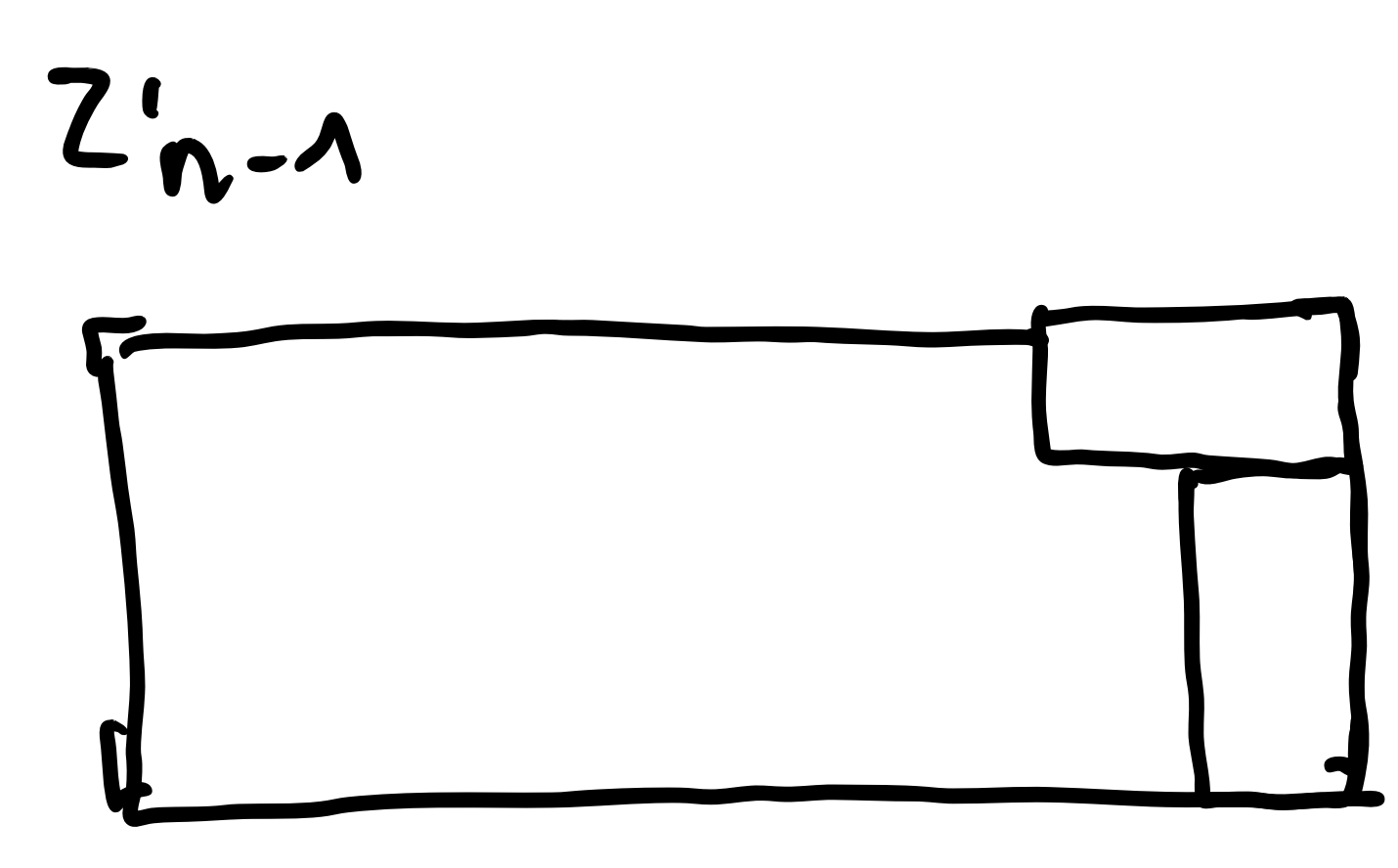
// Rq z_i : i correspond au nbr de colonne (tjrs 3 lignes).

A_1 : 0 pavages

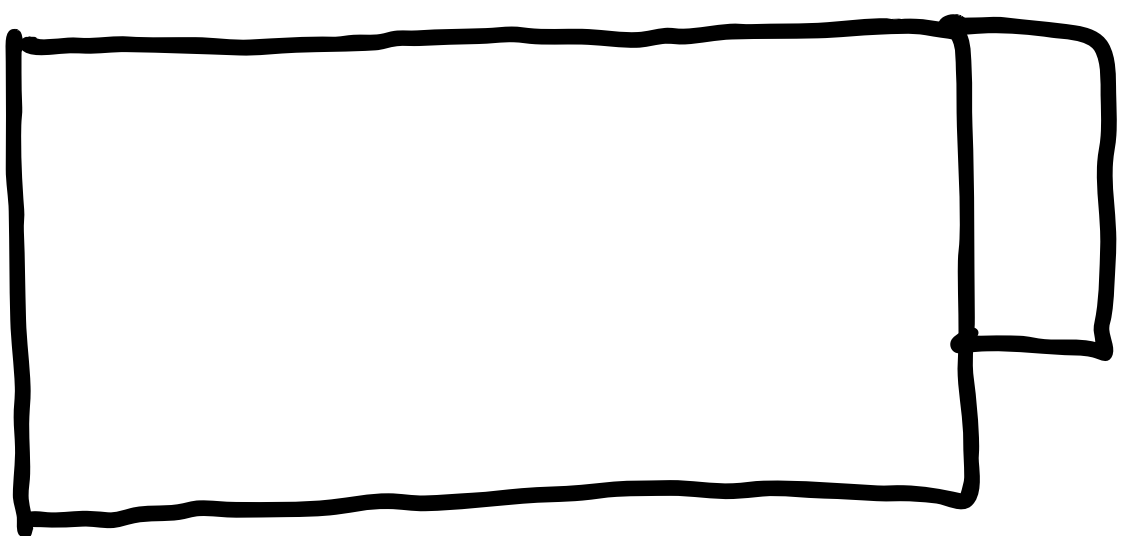
B_1 : 1 pavage

A_2 : 3 pavages

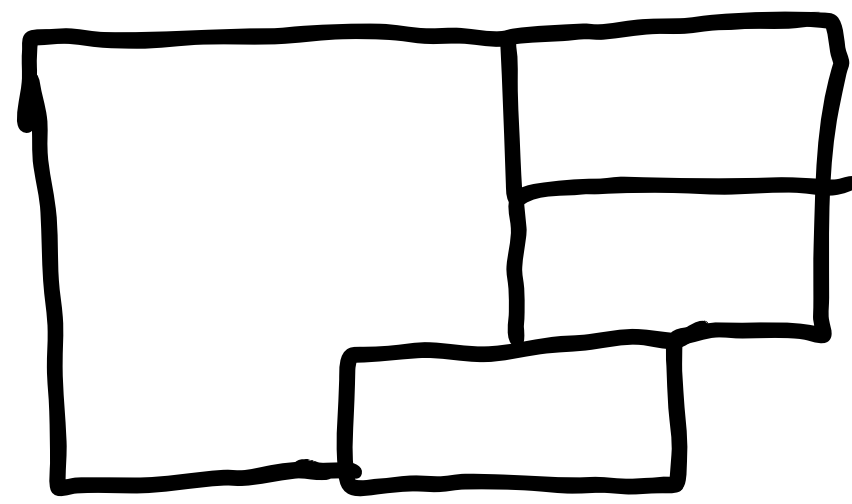
B_2 : 0 pavages



2^{n-1}



2^{n-2}



Nb Parages (n):

$A_1, B = 1$

A_0

B_1

Si n impair Alors retourner 0

Pour $i = 2$ à n

Si i pair: Alors $A_i = A + 2B$

¶ sinon: $B := A + B$

Retourner A