Schéma approximation polynomial

- on considère prob d'optimisation max/min
- un schéma d'approximation en tps polynomial - algo - :

input : $\begin{cases} \text{une instance du prob de taille } n \\ \text{un paramètre } \varepsilon > 0 \end{cases}$

output : un résultat $\geq (1-\varepsilon) \cdot K_{opt}$ ← sol opti

complexité : tps polynomial de n

// calcul approximatif temps polynomial
• Complexité peut être sous la forme $(2^{\varepsilon} \cdot n)$

efficace

Changer la complexitée ⇝ ∃ 2 autres critères.

Ex : FPTAS pour pb ~~sac~~ à dos

$\{ \quad O(V_n)^{\sum_{i=1}^{n} v_i}$

$T[i,v]$ = poids minimal pr obtenir la val v avec les objets $1, 2, \ldots, i$

$$T[i,v] = \begin{cases} \min\left(T[i-1,v],\ w_i + T[i-1, v-v_i]\right) & \text{si } v \geq v_i \\ T[i-1,v] & \text{sinon} \end{cases}$$

$T[0,v] = +\infty \qquad T[i,0] = 0$

Sol : cherche plus grand v tq $T[n,v] \leq W$

Application :

w	4	2	3
v	3	1	2

$W = 5$

$V = 3+1+2 = 6$

objet	0	1	2	3	4	5	6
0	0	+∞	+∞	+∞	+∞	+∞	+∞
1	0	+∞	+∞	4	+∞	+∞	+∞
2	0	2	+∞	4	6	+∞	+∞
3	0	2	3	4 ⑤	6	7	9

+∞ car prob min, de tt manière on va oublier ces vals.

+3 min

⑤ val max pouvant aller dans le sac
// pour la trv cherche sur dernière ligne de
// droite à gauche.

Algo Approx(E, w, ε):

- Retirer les objets avec poids $> w$
- $n = |E|$
- $v_M = \max_i (v_i)$
- Pour $i=1$ à n: $\hat{v}_i = \lfloor v_i \times \underbrace{\frac{n}{\varepsilon v_M}}_{\text{tjrs} < 1.} \rfloor$
- Soit $\hat{E}$ les objets $\{(w_1, \hat{v}_1), \dots, (w_n, \hat{v}_n)\}$
- Renvoyer le résultat de l'algo en $O(n.\hat{V})$ appliqué à $\hat{E}$

ss ensemble d'objets $\qquad \sum_i \hat{v}_i$

Complexité :

$\hat{v}_i \leqslant \frac{n}{\varepsilon}$

donc $\hat{V} = \Sigma \hat{v}_i \leqslant \frac{n^2}{\varepsilon} \rightsquigarrow O\left(\frac{n^3}{\varepsilon}\right)$

/pr savoir si optimal montrer: $K_{algo} \geqslant (1-\varepsilon) K_{opt}$

En gros :

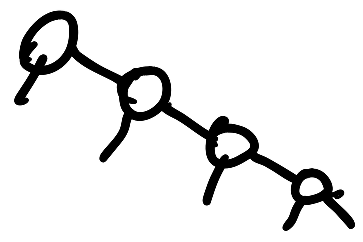
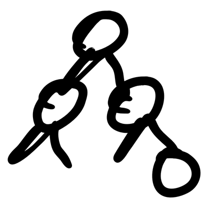
$$\begin{bmatrix} w: & - & - & - \dots \\ v: & 13042508 & 25013307 5 & --- \end{bmatrix}$$

$$\begin{bmatrix} w: & - & - & -- \\ v: & 208 & 400 & -- \end{bmatrix}$$

/réduction $\rightsquigarrow$ réduit les bornes /précision.

(ex: si 2 val de base sont != +-1 $\rightsquigarrow$ peuvent être $=$)

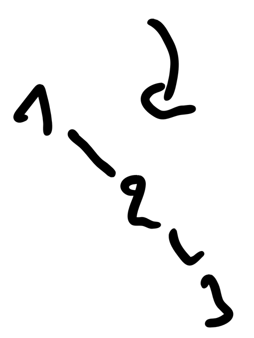
ABR optimaux :

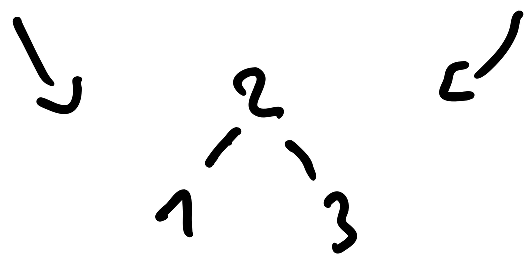
l'insertion d'el peut être problématique et former des peignes.  et non 

→ qlqs sol existe abr rouge/noir

mais de manière générale pas la peine, les bon abr apparaissent plus souvent

ex 1 2 3   2 1 3   2 3 1

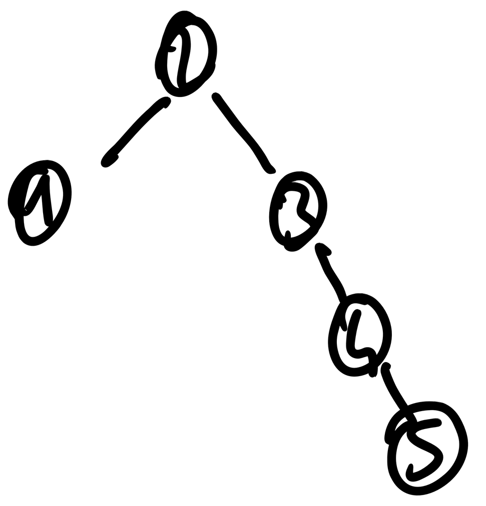


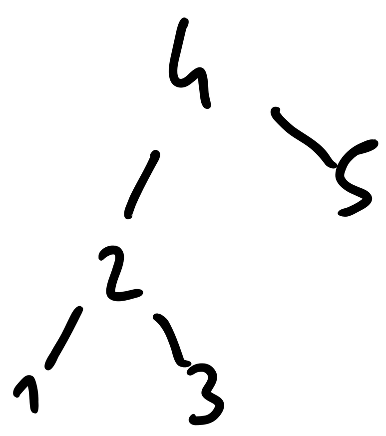


Algo pour obtenir ABR optimal (obtenir coût recherche + faible possible) :

x	1	2	3	4	5
h	10	10	10	50	10

Coût :

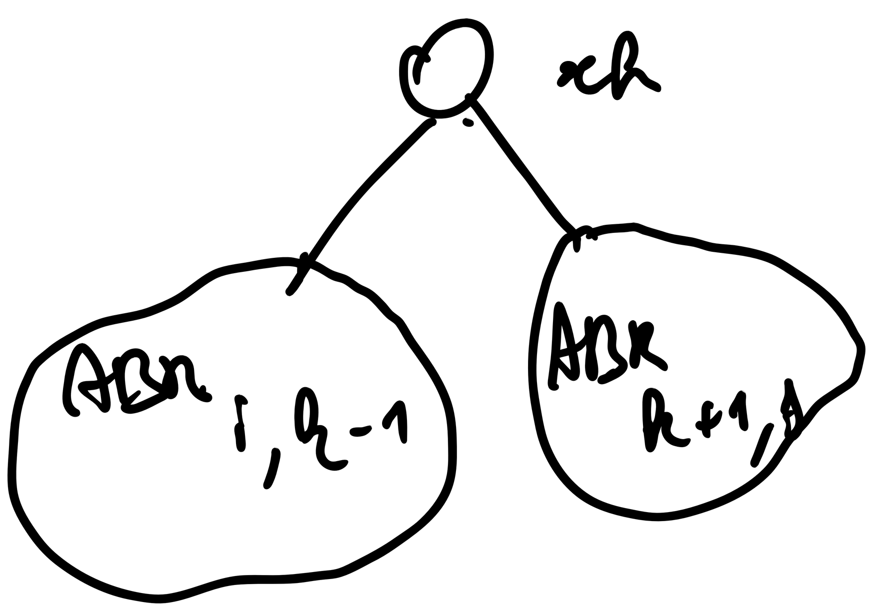




$10 + 2\times 10 + 2\times 10$
$+ 3\times 50 + 4\times 10$
$= 240$

$1\times 50 + 2\times 10 + 2\times 10$
$+ 3\times 10 + 3\times 10$
$= 150$

structure d'un "ABR i,j" : (chaque ss arbre contient un ensemble d'élément plage de i-j)



$i \le k \le j$

- Master theoreme   a, b   cas 1, cas 2 & cas 3
- tri algo diviser pr regner
- prog dynamique
- prog dynamique trv rec