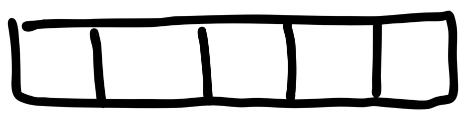
Compteur binaire k  taille k

opération "+1"

pire cas : $O(k)$

donc "à la louche" : n opérations +1 $\rightsquigarrow O(nk)$

```
increment (c)
    i = 0
    while i < k ∧ c[i] == 1
        c[i] = 0
        i++
    si i < k alors c[i] = 1
```

$k=4$

0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
...

agrégation méthode (1)

coût tt de n increments = le nbr total de bascules d'un bit de $0 \to 1$ ou $1 \to 0$

c[0] : change à chq opération +1

c[1] : change un increment sur 2

c[2] : change un increment sur 4

---

coût total de "n +1" : $\sum_{i=0}^{k-1} \lfloor \frac{n}{2^i} \rfloor < n \underbrace{\sum_{i=0}^{k-1} \frac{1}{2^i}}_{<2}$

↳ $O(n)$

coût amorti de l'opération "+1" : $O(1)$

Méthode comptable : ②) // plus intuitive

On paie 2 crédits par un +1

→ 1 crédit sert à passer un 0 à 1

→ 1 crédit est stocké avec ce "nouveau" 1 et servira lorsqu'il repassera à 0 (plus tard

$\hat{C}_i = 2$

⇝ La complexité totale est bien $O(n)$

Méthode du potentiel ③) // mécanique

Coût réel du i-ème +1 $= t_i + 1$

$\Phi(C)$ = Le nb de 1 dans la valeur du cpteur (en binaire)

$$\boxed{\hat{C}_i = \underbrace{C_i}_{t_i+1} + \underbrace{\Phi(C_i)}_{b_i} - \underbrace{\Phi(C_{i-1})}_{b_{i-1}}}$$

C_i = cpteur avec le ième +1

b_i = le nb de "1" dans C_i

t_i = le nb de 1 qui passent à 0 lors du i-ème incrément

Évaluer $b_i - b_{i-1}$?

→ Si $b_i = 0$: il y a eu un reset : $b_{i-1} = k$, $t_i = k$ → $b_i = b_{i-1} - t_i$ ($t_i \geq 0$)

$\hookrightarrow k \leq t_i \leq 0$

→ Si $b_i > 0$: on a : $b_i = b_{i-1} - t_i + \underbrace{1}_{\text{le nouveau 1}}$

on a tjrs : $b_i - b_{i-1} \leq 1 - t_i$

Donc :

$\hat{C}_i \leq \underbrace{C_i}_{t_i+1} + 1 - t_i = 2$

. $\hat{C}_i$ est en $O(1)$

. n "+1" en $O(n)$

Tableaux dynamiques

Structure de données de type "tableaux" : zone mémoire avec adressage direct

2 opérations : { . ajouter
. supprimer   (+ accès aux éléments)

T.nb : nb d'éléments stockés dans T   // nb ≤ taille.
T.taille : nb de "slots" actuellement réservés pour T (= capacité max courante)

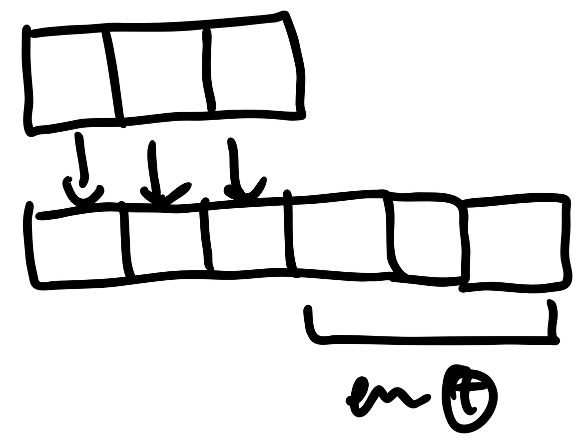
→ l'opération Ajouter peut obliger à recopier tous les éléments stockés dans une autre zone mémoire

Ajouter (T, x)

```
Si T.taille == 0 Alors | T = new Table de taille 1
                       | T.taille = 1

Si T.nb == T.taille Alors :
    | T' = new Table de taille 2*T.taille
    | T'.taille = 2 taille
    | T'.nb = T.nb
    | Recopier T dans T'
    | (libérer la zone mem de T)
    | T = T'

T[nb] := x
T.nb ++
```

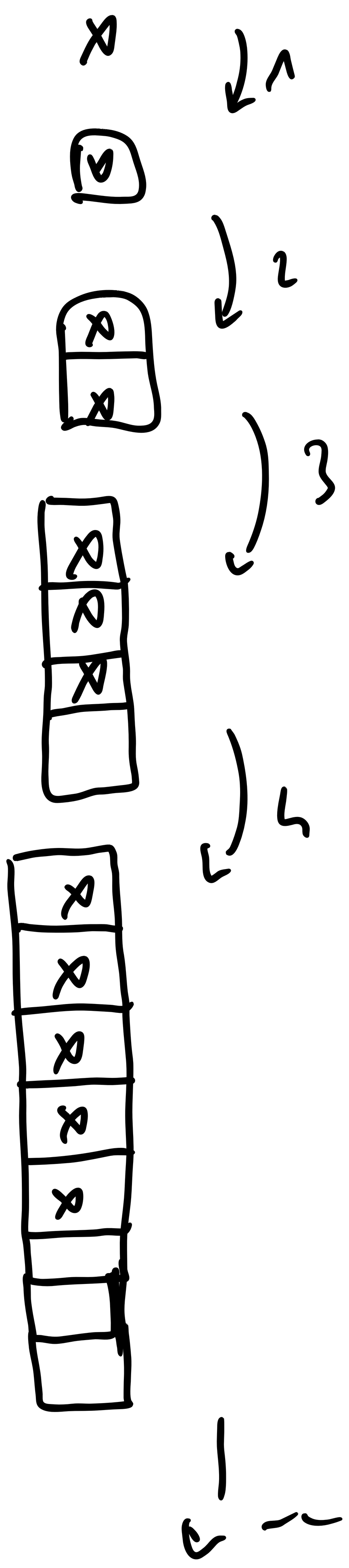


↳ $O(n^2)$ dans n opérations ajouter

Supprimer (T):
si T.nb > 0
| T.nb

Méthode de l'agrégation

Coût réel du i-ème ajout : $c_i = \begin{cases} i & \text{si } i-1 = 2^a \\ 1 & \text{sinon} \end{cases}$



Coût cumulé de n ajouts :

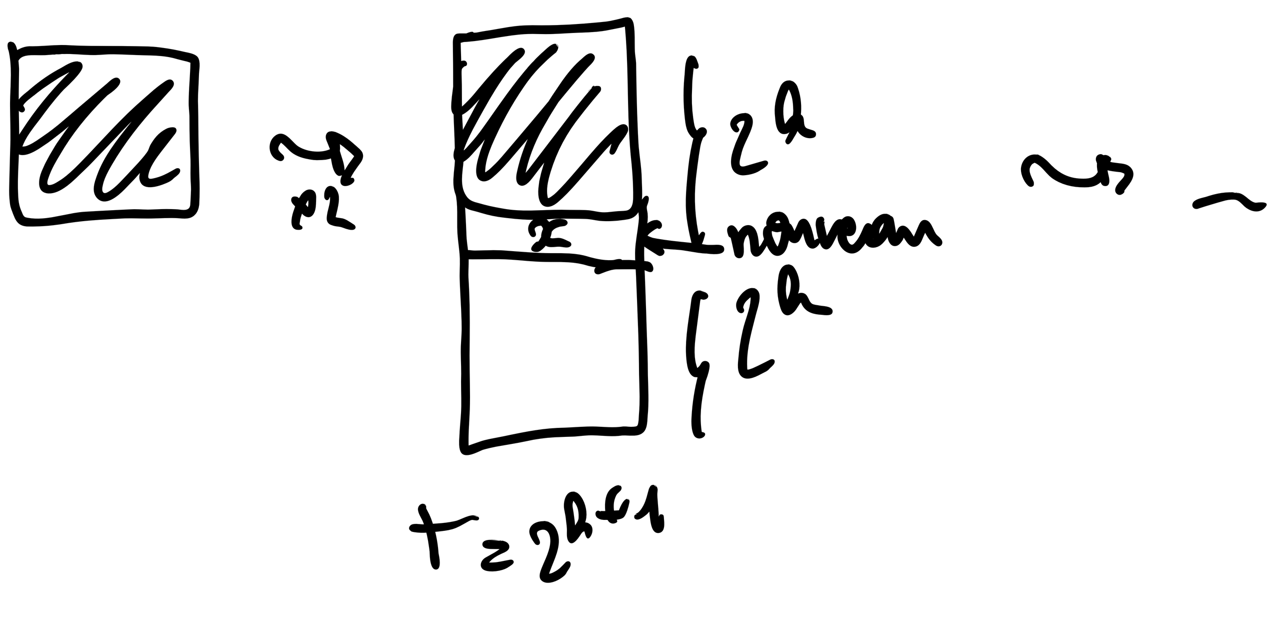
$$\underbrace{\sum_{i=1}^{n} c_i \leq n + \underbrace{\sum_{j=1}^{\lfloor \log n \rfloor} 2^j}_{\leq 2n}}_{\leq 3n}$$

Coût amorti de « Ajouter » = 3

Méthode comptable :

Coût amorti = 3

1 crédit pour insérer l'élément
1 ——— recopier
1 ——— recopier un des "vieux" de T) à la prochaine expansion... si besoin



Def : taux de remplissage du tab

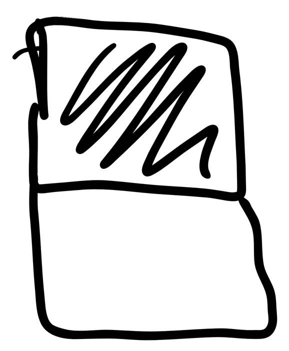
Méthode du potentiel

$\Phi(T) = 2 \cdot T.nb - T.taille$

Remarques :

1) $\Phi(T) = 0$ après chaque expansion →

2) Juste avant une expansion, $\Phi(T) = T.nb$ →



3) Le tableau est tjrs entre la moitié plein et plein

$$\frac{1}{2} \le \alpha(T) \le 1$$

et donc $\Phi(T) \ge 0$ à tt moment

Évaluation du coût amorti de ajouter :

On considère la i-ème opération ajouter : 2 cas :

▷ cet ajout n'entraîne pas une expansion

$$\hat{c}_i = c_i + \Phi(T_i) - \Phi(T_{i-1})$$
$$1 + (2 \cdot T.nb_i - T.taille_i) - (2 \cdot T.nb_{i-1} - T.taille_{i-1})$$
$$1 + (2 \cdot nb_i - t_i) - (2\, nb_{i-1} - t_{i-1})$$

$$\begin{cases} nb_i = nb_{i-1} + 1 \\ t_i = t_{i-1} \end{cases}$$

$$\hat{c}_i = 1 + 2\,nb_{i-1} + 2 - t_i - 2\,nb_{i-1} + t_i$$
$$= 3$$

▷ cet ajout entraîne une expansion de T :

$$\hat{c}_i = \underbrace{c_i}_{\underbrace{nb_{i-1}}_{\text{recopie}} + \underbrace{1}_{\text{nouvel elt}}} + \underbrace{\Phi(T_i)}_{(2nb_i - t_i)} - \underbrace{\Phi(T_{i-1})}_{(2nb_{i-1} - t_{i-1})}$$

Def : taux de remplissage du tab

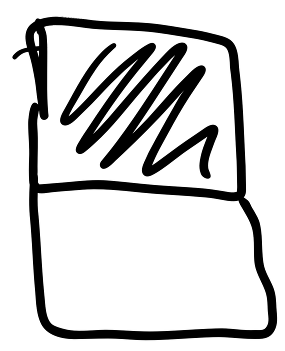
Méthode du potentiel

$$\Phi(T) = 2 . T.nb - T.taille$$

Remarques :

1) $\Phi(T) = 0$ après chaque expansion $\rightarrow$

2) Juste avant une expansion, $\Phi(T) = T.nb \rightarrow$



3) Le tableau est tjrs entre la moitié plein et plein

$$\frac{1}{2} \leq \alpha(T) \leq 1$$

et donc $\Phi(T) \geq 0$ à tt moment

Evaluation du coût amorti de ajouter :

On considère la i-ème opération ajouter : 2 cas :

▷ Cet ajout n'entraîne pas une expansion

$$\hat{c}_i = c_i + \Phi(T_i) - \Phi(T_{i-1})$$

$$1 + (2 . T.nb_i - T.taille_i) - (2 . T.nb_{i-1} - T.taille_{i-1})$$

$$1 + (2 . nb_i - t_i) - (2\, nb_{i-1} - t_{i-1})$$

$$\begin{cases} nb_i = nb_{i-1} + 1 \\ t_i = t_{i-1} \end{cases}$$

▷ $\hat{c}_i = 1 + 2\, nb_{i-1} + 2 - t_i - 2\, nb_{i-1} + t_i$

$= 3$

▷ cet ajout entraîne une expansion de T :

$$\hat{c}_i = \underbrace{c_i}_{\underbrace{nb_{i-1}}_{\text{recopie}} + \underbrace{1}_{\text{nouvel élét}}} + \underbrace{\Phi(T_i)}_{(2nb_i - t_i)} - \underbrace{\Phi(T_{i-1})}_{(2nb_{i-1} - t_{i-1})}$$

$$\begin{cases} nb_i = nb_{i-1} + 1 \\ t_i = 2 \cdot t_{i-1} \end{cases}$$

et $nb_{i-1} = t_{i-1}$ (recopie ...)

$$\hat{C}_i = (nb_{i-1} + 1) + (2\,nb_{i-1} + 2 - 2t_{i-1}) - (2nb_{i-1} - t_{i-1})$$

$$= nb_{i-1} + 3 - t_{i-1}$$

$$= 3$$