

Méthode comptable

Posons que le coût amortie $ca(i)$ pour empiler vaut 2 crédits et le reste valent 0.

C'est à dire on paye un crédit et on garde un crédit.

Résumé

Opération	coût réel	coût amortie
Insertion	1	2
Dépilement	1	0
Multidépilement	$\min(k,n)$	0

Donc, la complexité est belle et bien $\Theta(1)$ amortie.

Méthode du potentiel

Posant $\Phi(D_i) = p$ où p représente le nombre d'éléments dans la pile.
Empiler :

$$\begin{aligned} ca(i) &= 1 + p - (p - 1) \\ &= 1 + 1 = 2 \end{aligned}$$

Depiler :

$$\begin{aligned} ca(i) &= 1 + (p - 1) - p \\ &= 1 - 1 = 0 \end{aligned}$$

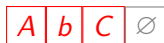
MultiDepiler :

$$\begin{aligned} ca(i) &= k + (p - k) - p \\ &= k - k = 0 \end{aligned}$$

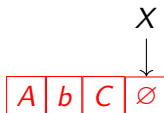
Pour chaque opération, si on effectue une séquence de n opérations, le coût amortie total est inférieur ou égal à $2n$. Donc, chaque opération est effectuée en $\Theta(1)$ amortie.

Intuitivement

Supposons qu'on veut insérer un élément nommé X dans le tableau suivant :

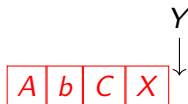


Alors le coût d'insertion sera de 1, car il reste une place dans le tableau :

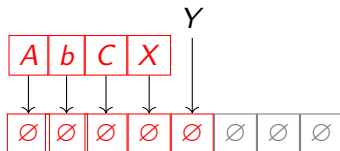


Qu'est ce que si passe si on insère un nouvel élément Y dans le tableau après insertion de X ?

Intuitivement

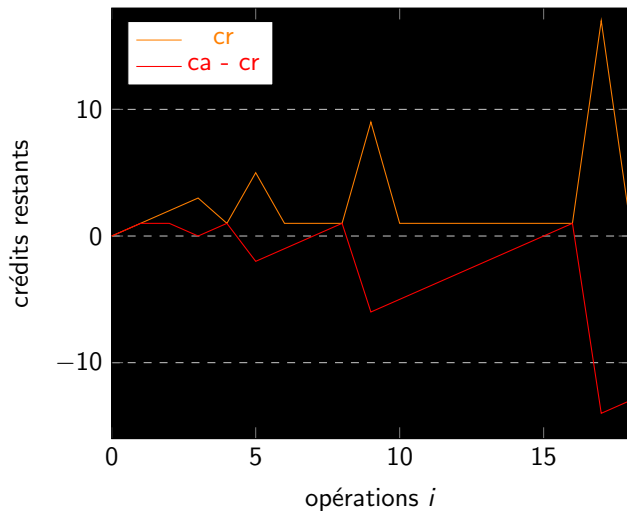


Solution : crée un tableau plus grand que le précédent, copier les éléments de l'ancien tableau dans le nouveau et ensuite mettre le nouvel élément dans le nouveau.

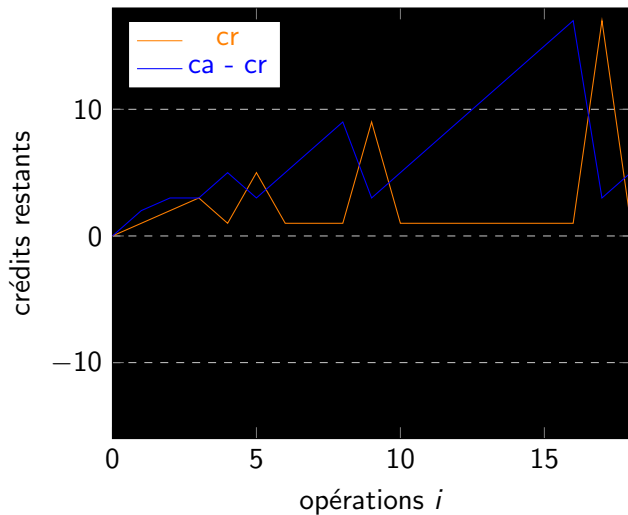


Cela nous a coûté 5 insertions (4 pour transférer les anciens dans le nouveau tableau et 1 pour insérer le nouvel élément) pour ce cas-ci.

Visuellement



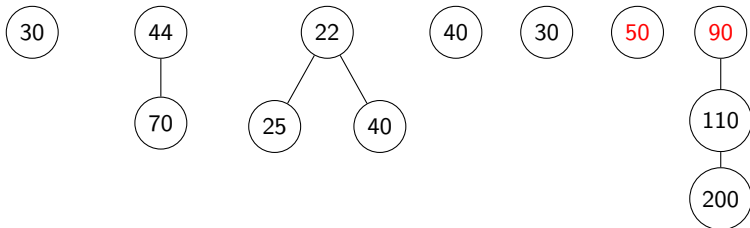
Visuellement



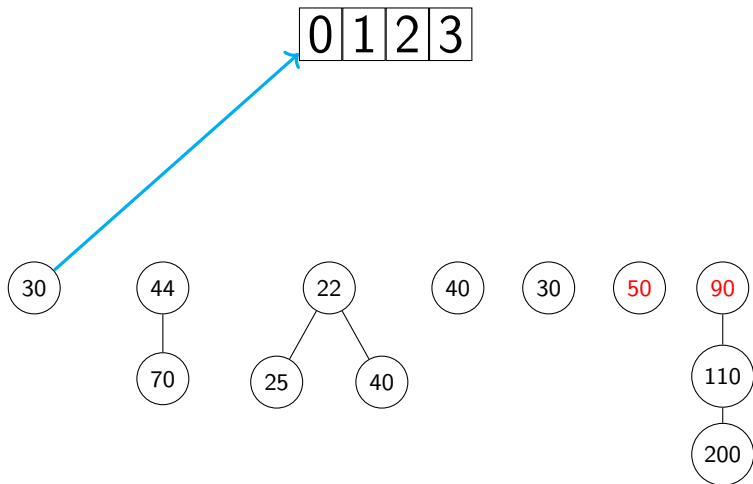
Exemple

En reprenant notre exemple de tout à l'heure, notre tableau doit être de taille 4, ce qui donne le tableau suivant :

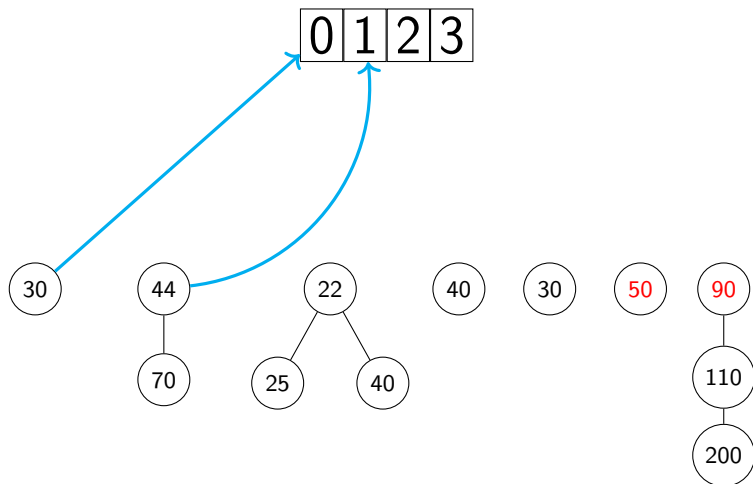
0	1	2	3
---	---	---	---



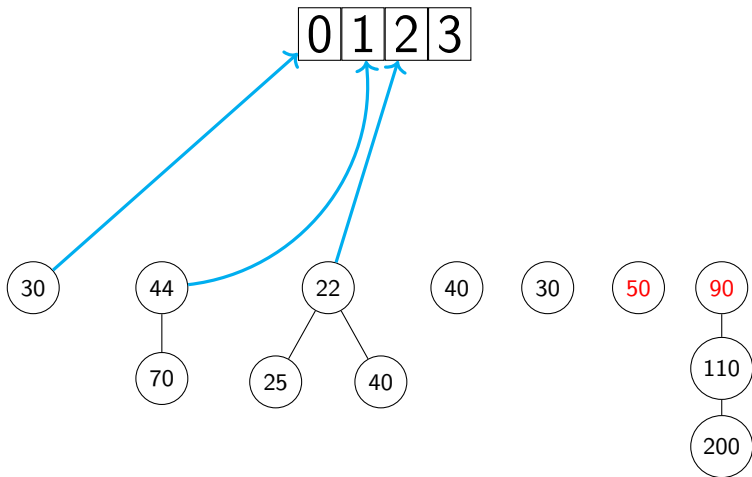
Racine 1



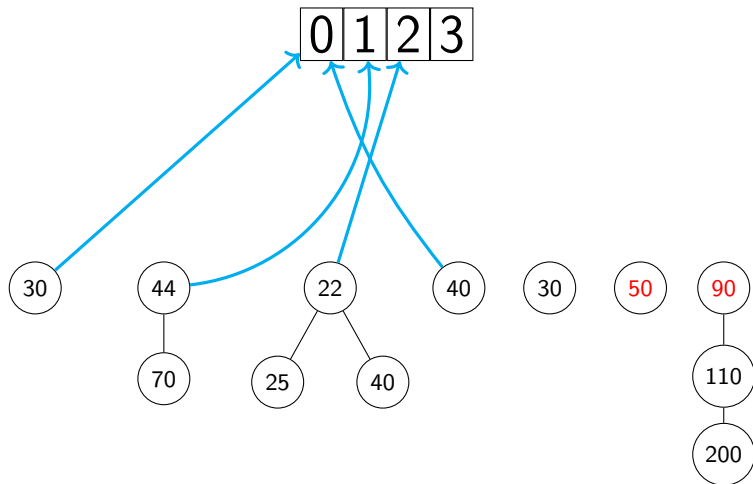
Racine 2



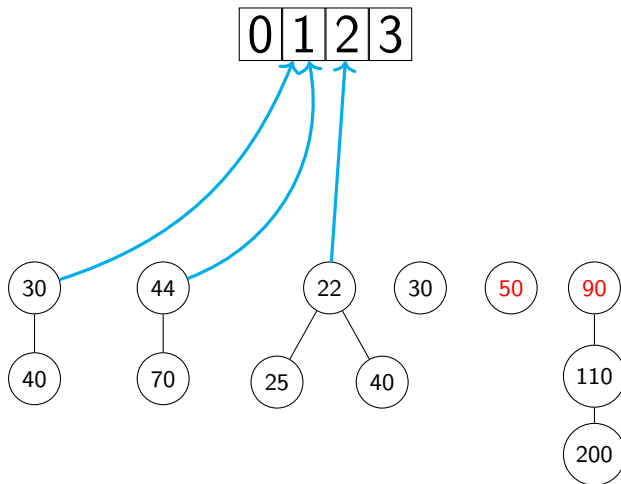
Racine 3



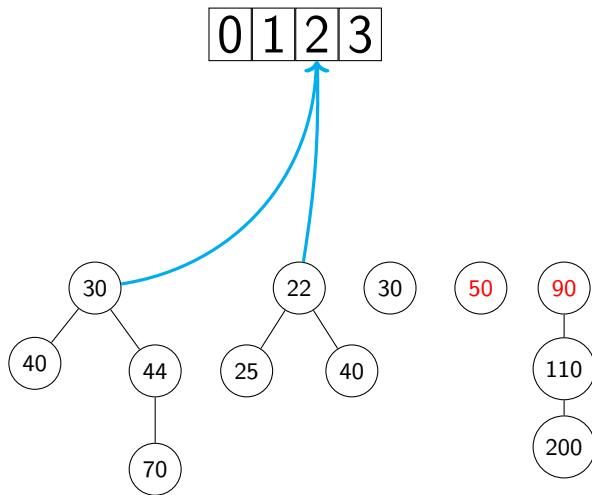
Racine 4



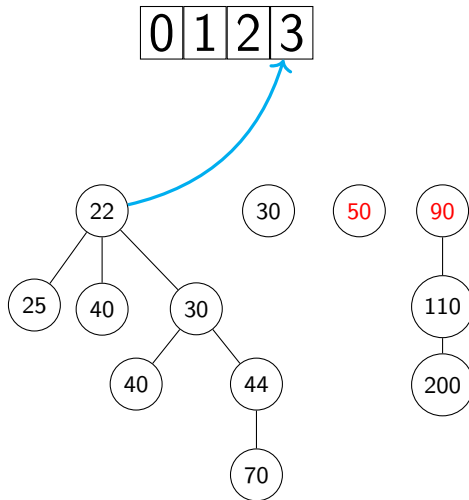
Fusion 1



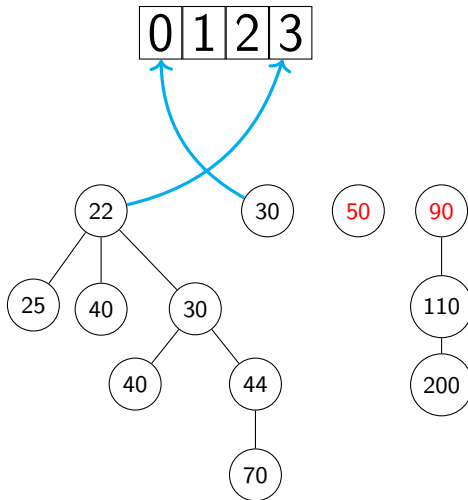
Fusion 2



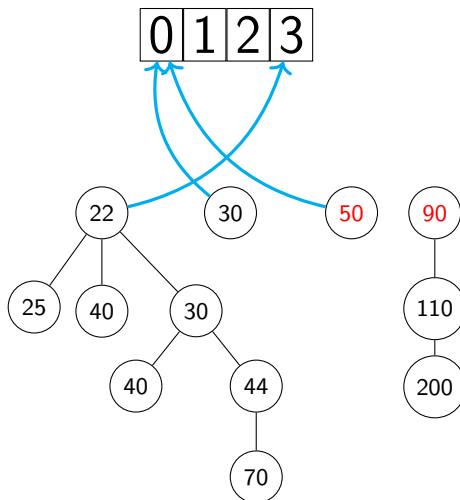
Fusion 3



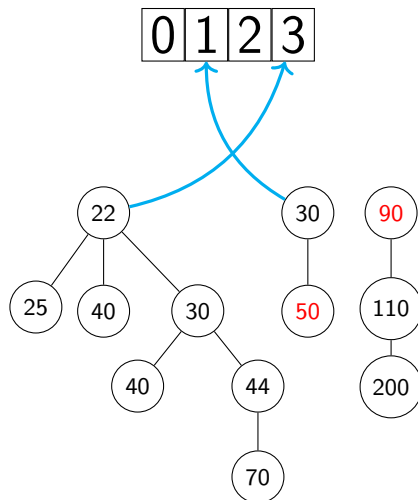
Racine 5



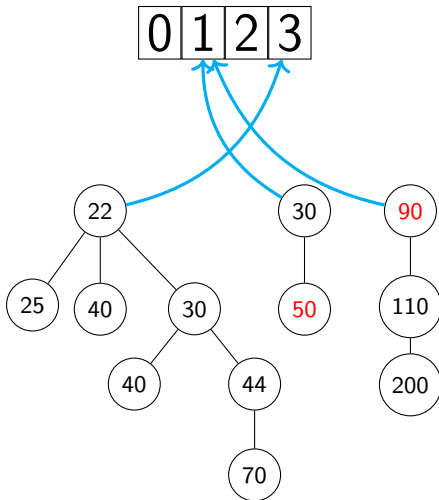
Racine 6



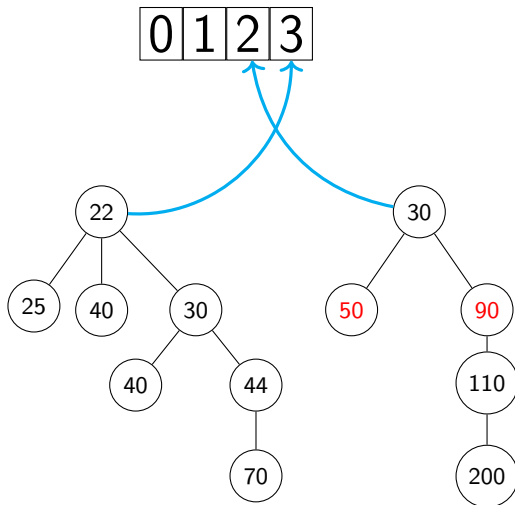
Fusion



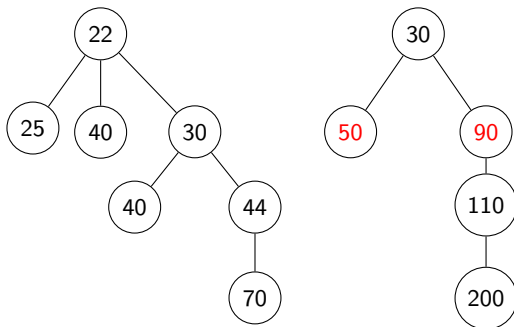
Racine 7



Fusion



État final



Et voilà !!

Act 3

Dans le monceau de Fibonacci, après la deuxième étape, on cherche le minimum.

Coût

Le coût total de l'opération peut être représenté de la manière suivante :

- 1 Étape 1 : $\Theta(\text{degré du minimum})$
- 2 Étape 2 : $\Theta(\text{degré maximale} + \#arbres)$
- 3 Étape 3 : $\Theta(\text{degré maximale})$

Ok, mais que vaut le degré maximale ? Que se passe t'il si on enlève le minimum avec juste des racines de degré 0 ?

Intuition

Ce qu'on remarque c'est qu'on fusionne deux racines parce qu'ils ont le même degré. Une fois la fusion faite, la nouvelle racine augmente son degré de 1 et la somme de celle-ci des éléments des deux racines fusionnées est de $\leq 2^d$ où d est le degré des deux racines qui ont été fusionnées.

- degré 0 : 1

Intuition

Ce qu'on remarque c'est qu'on fusionne deux racines parce qu'ils ont le même degré. Une fois la fusion faite, la nouvelle racine augmente son degré de 1 et la somme de celle-ci des éléments des deux racines fusionnées est de $\leq 2^d$ où d est le degré des deux racines qui ont été fusionnées.

- degré 0 : 1
- degré 1 : 2

Intuition

Ce qu'on remarque c'est qu'on fusionne deux racines parce qu'ils ont le même degré. Une fois la fusion faite, la nouvelle racine augmente son degré de 1 et la somme de celle-ci des éléments des deux racines fusionnées est de $\leq 2^d$ où d est le degré des deux racines qui ont été fusionnées.

- degré 0 : 1
- degré 1 : 2
- degré 2 : 4

Intuition

Ce qu'on remarque c'est qu'on fusionne deux racines parce qu'ils ont le même degré. Une fois la fusion faite, la nouvelle racine augmente son degré de 1 et la somme de celle-ci des éléments des deux racines fusionnées est de $\leq 2^d$ où d est le degré des deux racines qui ont été fusionnées.

- degré 0 : 1
- degré 1 : 2
- degré 2 : 4
- degré 3 : 8

Intuition

Ce qu'on remarque c'est qu'on fusionne deux racines parce qu'ils ont le même degré. Une fois la fusion faite, la nouvelle racine augmente son degré de 1 et la somme de celle-ci des éléments des deux racines fusionnées est de $\leq 2^d$ où d est le degré des deux racines qui ont été fusionnées.

- degré 0 : 1
- degré 1 : 2
- degré 2 : 4
- degré 3 : 8
- degré 4 : 16

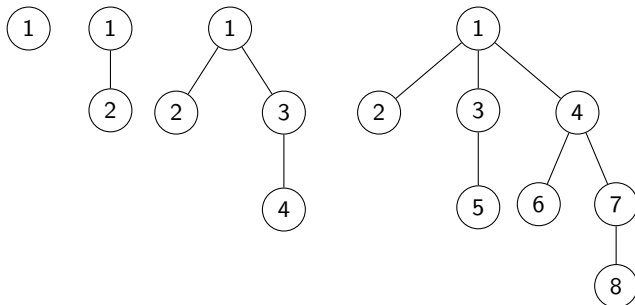
Intuition

Ce qu'on remarque c'est qu'on fusionne deux racines parce qu'ils ont le même degré. Une fois la fusion faite, la nouvelle racine augmente son degré de 1 et la somme de celle-ci des éléments des deux racines fusionnées est de $\leq 2^d$ où d est le degré des deux racines qui ont été fusionnées.

- degré 0 : 1
- degré 1 : 2
- degré 2 : 4
- degré 3 : 8
- degré 4 : 16

Note : on suppose qu'on fait pas de réduction de clés.
Pas convaincu ?

Intuition visuelle



Remarque : ces arbres sont des arbre binomiaux et ils augmentent de façon exponentielle. Soit d_{max} le degré maximale d'un arbre à n noeuds, alors $\log_2 n = d_{max}$.

degré maximale : $\log \# \text{nombre de clés} \in \mathcal{O}(\log n)$

Modification de priorité de clé

Supposons qu'on a un accès direct sur une clé (table de hachage), la suppression d'une clé se fait comme suit :

- Changer la clé avec la nouvelle priorité.

Modification de priorité de clé

Supposons qu'on a un accès direct sur une clé (table de hachage), la suppression d'une clé se fait comme suit :

- Changer la clé avec la nouvelle priorité.
- Si la clé modifié est toujours plus grand que son parent, alors on ne fait rien. Sinon, on doit couper ce noeud (avec ses enfants) et le mettre dans la liste des racines. Mettre à jour le min si nécessaire.

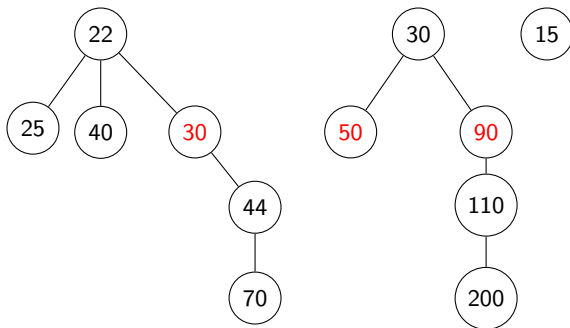
Modification de priorité de clé

Supposons qu'on a un accès direct sur une clé (table de hachage), la suppression d'une clé se fait comme suit :

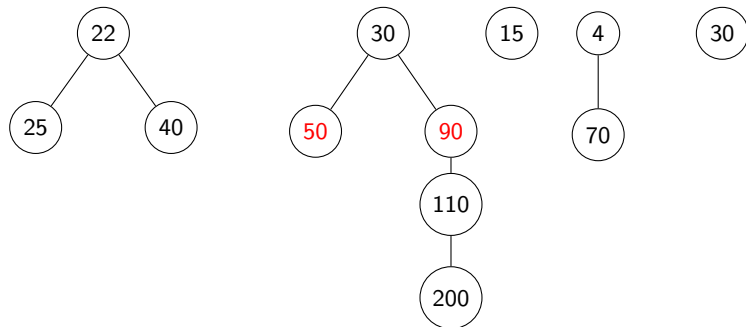
- Changer la clé avec la nouvelle priorité.
- Si la clé modifié est toujours plus grand que son parent, alors on ne fait rien. Sinon, on doit couper ce noeud (avec ses enfants) et le mettre dans la liste des racines. Mettre à jour le min si nécessaire.
- Si le parent a perdu un enfant, alors il est marqué. S'il est déjà marqué, alors à son tour il se fait mettre dans liste des racines et on enlève sa marque. On refait le même processus jusqu'à atteindre la racine de l'arbre modifié ou jusqu'à atteindre un parent non marqué.

Exemple ($40 \rightarrow 15$)

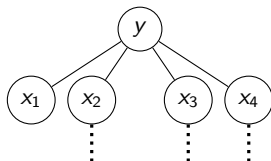
En reprenant toujours le même arbre après suppression du minimum, réduisons les clé 40 (rattaché à 30) pour 15 et 44 pour 4 :



Réduction clé 44 \rightarrow 4



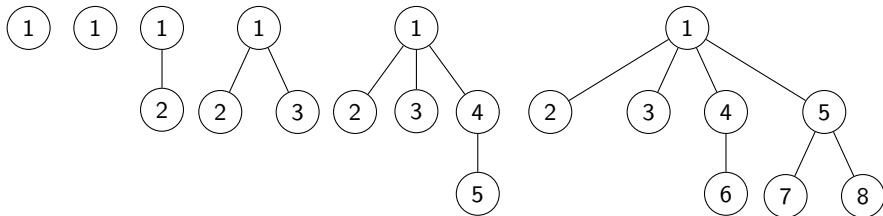
Intuition



Pour que x_4 soit fusionner avec y , y était de degré 3 et x_4 de degré 3. Si x_4 perd un enfant et il n'est pas marqué, alors il sera marqué et son degré descendra de 1. Si x_4 perd un autre enfant et qu'il a déjà été marqué, alors il doit être enlever de cet arbre et son degré est décrémenter de 1 et il x_4 devient une nouvelle racine. De plus, le degré de y est aussi décrémenté de 1 dû que x_4 n'est plus un enfant de y .

Donc, le degré de x_4 en tant que nouvelle racine est au moins nouveau degré $\geq d_4 - 2$. Construisons la taille minimale que les arbres auront avec cette règle.

Minimum



Remarquez-vous la séquence ? Suite de Fibonacci !!

Si on fait le rapport entre un nombre de Fibonacci et son suivant, on obtient le résultat suivant :

$$\lim_{n \rightarrow \infty} \frac{F_{n+1}}{F_n} = \varphi = \frac{1 + \sqrt{5}}{2} \approx 1,618$$

où F_i est inième terme de la suite de Fibonacci. Fait intéressant : $\varphi^2 = \varphi + 1$.

Nombre de noeuds dans un arbre

Soit, d le degré de l'arbre d'un monceau et n son nombre de noeuds, alors le nombre de noeud est le suivant :

$$F_{d+2} \leq n$$

Note : on peut démontrer que $n \geq \varphi^d$ ou dans ce cas-ci $F_{d+2} \geq \varphi^d$.

Analyse

Combien de noeuds sont au maximum coupés ? On coupe soit un noeud parce que la modification de clé n'est pas respecté entre la clé modifié ou que le parent a déjà été marqué.

En d'autre termes, le nombre de réduction de clés k engendre au plus $2k$ coupures de noeuds. Donc, $2k$ futures racines.

