Arbres splay

Réf: Kozen : The design and analysis of algorithms

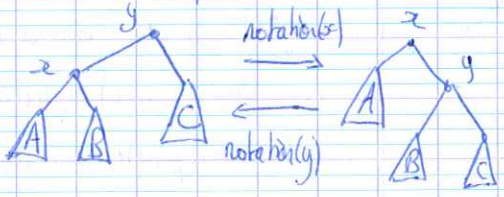
Les arbres splay sont des arbres binaires de recherche

- Les arbres splay permettent de faire les opérations suivantes en coût amorti $O(\log n)$:
  - membre (i,S) : détermine si i est un élément de l'arbre S
  - ajout (i,S) : ajoute i dans S s'il n'est pas présent
  - supp (i,S) : supprime i de S s'il est présent
  - union (S,S') : fusionne S et S' dans un seul arbre splay, avec comme hypothèse $S < S'$ (i.e. $\forall x \in S, \forall y \in S', x < y$)
  - pivote (i,S) : coupe l'arbre S en deux arbres splay S' et S'' de sorte que $x \leq i \leq y$ pour tout $x \in S'$ et $y \in S''$

- Toutes ces opérations se font à l'aide d'une opération splay :
  splay (i,S) : réorganise l'arbre splay S de sorte que l'élément i soit à la racine si $i \in S$, et sinon la racine est $\max \{k \in S \mid k < i\}$

Ainsi, on peut faire :
- membre (i,S) : splay (i,S) et on teste la racine
- ajout (i,S) : splay (i,S) et on insère i à la racine
- supp (i,S) : splay (i,S) et on supprime la racine et on appelle union
- union (S,S') : on fait splay $(+\infty, S)$ et on met S' en fils droite de la racine
- pivote (S,i) : splay (i,S) et on récupère les fils de la racine

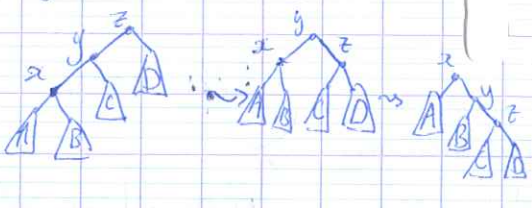
- L'opération splay se fait à l'aide d'une opération élémentaire rotation assez classique dans les ABR :



on voit que cette opération fait remonter x vers la racine, mais il faut faire un peu plus attention si on veut que l'arbre soit équilibré
(i) si x est fils de la racine : zig (x) = on fait rotate (x)

(droite)

(ii) Si x et son père y sont tous les deux fils gauches : $zig\text{-}zig(x)$ = on applique rotation(y) puis rotation(x) :



(iii) si x et son père y sont fils de côté différents : $zig\text{-}zag(x)$ = on applique rotation(x) deux fois

- Analyse amortie de la complexité de splay :

On définit, pour x un noeud de S, $S(x)$ = le sous-arbre enraciné à x.

On définit : $\begin{cases} \mu(S) = \lfloor \log |S| \rfloor \\ \mu(x) = \mu(S(x)) \end{cases}$

Invariant Noeud x a au moins $\mu(x)$ crédits.

Lemme : une opération splay(x,S) nécessite pas plus de $3(\mu(S) - \mu(x)) + 1$ crédits pour effectuer l'opération et maintenir l'invariant

Démo On regarde dans quel cas on est pour x :

- (i) zig : on remarque que $\mu'(x) = \mu(y)$ et $\mu'(y) \leq \mu'(x)$ (x a y fils de x), d'où un coût de $\mu'(x) + \mu'(y) - \mu(x) - \mu(y) = \mu'(y) - \mu(x)$ $\quad \mu(S)$
$\leq \mu'(x) - \mu(x) \leq 3(\mu'(x) - \mu(x))$

et on a un extra-crédit pour gérer les opérations en $O(1)$

- (ii) et (iii) : on va montrer que pour une étape le coût est $\leq 3(\mu'(x) - \mu(x))$ et donc en sommant sur la remontée, on aura une somme télescopique et c'est gagné

* (ii) : zig-zig : on a besoin de $\mu'(x) + \mu'(y) + \mu'(z) - \mu(x) - \mu(y) - \mu(z) = : E$

on a $\mu'(x) = \mu(z)$ et on a que μ est croissante suivant la relation père-fils, d'où

$E = \mu'(x) + \mu'(y) + \mu'(z) - \mu(x) - \mu(y) - \mu(z) = \mu'(y) + \mu'(z) - \mu(x) - \mu(y)$
$\leq \mu'(x) + \mu'(x) - \mu(x) - \mu(x) = 2(\mu'(x) - \mu(x))$

et on peut payer les opérations sauf si $\mu'(x) = \mu(x)$

* Mais on montre que dans ce cas, on a $E < 0$. par l'absurde :

si $\mu'(x) = \mu(x) = \mu(z)$, alors $\mu(x) = \mu(y) = \mu(z)$

en supposant $E \geq 0$, on trouve de même $\mu'(x) = \mu'(y) = \mu'(z)$

Impossible par propriété du log

□