Module EA4 – Éléments d'Algorithmique II

Outils pour l'analyse des algorithmes

Dominique Poulalhon

`dominique.poulalhon@irif.fr`

Université Paris Cité

L2 Informatique & DL Bio-Info, Jap-Info, Math-Info

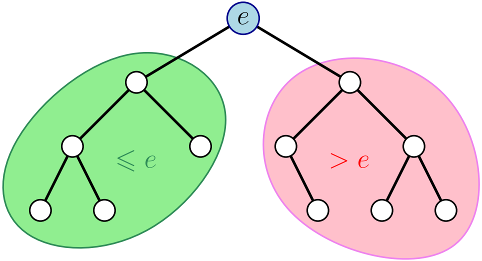
Année universitaire 2023-204

Conférence-débat « Les communs numériques » : aujourd'hui 16h-18h, amphi 10E

Partiel : vendredi 22 mars, 16h15-18h00, amphis 1A et 2A

Un arbre binaire de recherche (ABR) est un arbre binaire, étiqueté, tel que l'étiquette de ***chaque*** sommet est comprise entre

- ***toutes*** *les étiquettes du sous-arbre gauche (plus petites)* et
- ***toutes*** *les étiquettes du sous-arbre droit (plus grandes)*



Théorème

la liste triée des éléments d'un ABR de taille n peut être obtenue en temps $\Theta(n)$ par un parcours en profondeur infixe.

Théorème

la recherche, l'ajout et la suppression d'un élément dans un ABR de hauteur h se font en temps $\Theta(h)$ au pire.

Corollaire

la construction d'un ABR de taille n par insertion successive de ses éléments a un coût $O(nh)$, si h est la hauteur de l'arbre obtenu.

La clé de l'efficacité de ces opérations réside donc dans la hauteur de l'ABR considéré en fonction de sa taille.

la hauteur $h(A)$ d'un arbre binaire A à n sommets vérifie :

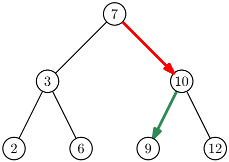
$$\log n \leqslant h(A) \leqslant n - 1$$

la hauteur $h(A)$ d'un arbre binaire A à n sommets vérifie :

$$\log n \leqslant h(A) \leqslant n - 1$$

et ces bornes sont atteintes :

Cas sympathique : ABR « parfait »



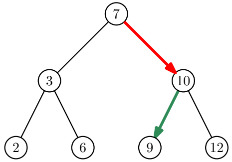
Hauteur $\log n$, recherche semblable à une recherche dichotomique, modifications efficaces, *i.e.* avantages du tableau trié sans les inconvénients

la hauteur $h(A)$ d'un arbre binaire A à n sommets vérifie :

$$\log n \leqslant h(A) \leqslant n - 1$$

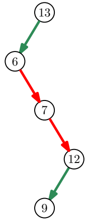
et ces bornes sont atteintes :

Cas sympathique : ABR « parfait »



Hauteur $\log n$, recherche semblable à une recherche dichotomique, modifications efficaces, *i.e.* avantages du tableau trié sans les inconvénients

Cas désagréable : ABR « filiforme »

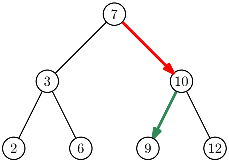


Hauteur $n - 1$, structure compliquée mais aussi inefficace qu'une liste chaînée non triée

la hauteur $h(A)$ d'un arbre binaire A à n sommets vérifie :
$$\log n \leqslant h(A) \leqslant n - 1$$

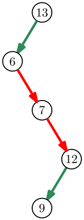
et ces bornes sont atteintes :

Cas sympathique : ABR « parfait »



Hauteur $\log n$, recherche semblable à une recherche dichotomique, modifications efficaces, *i.e.* avantages du tableau trié sans les inconvénients

Cas désagréable : ABR « filiforme »



Hauteur $n - 1$, structure compliquée mais aussi inefficace qu'une liste chaînée non triée

À quoi ressemble un ABR « typique » ? À un arbre binaire parfait ? À un arbre filiforme ? Ou ni l'un ni l'autre ?

Observation : chaque arbre binaire à n sommets admet un unique étiquetage par $\{1, \ldots, n\}$ respectant les contraintes d'un ABR

Observation : chaque arbre binaire à n sommets admet un unique étiquetage par $\{1, \ldots, n\}$ respectant les contraintes d'un ABR

Problème (?) :

Théorème (admis)

la hauteur moyenne d'un arbre binaire choisi uniformément parmi les arbres binaires à n nœuds est en $\Theta(\sqrt{n})$.

Observation : chaque arbre binaire à n sommets admet un unique étiquetage par $\{1, \ldots, n\}$ respectant les contraintes d'un ABR

Problème (?) :

Théorème (admis)

la hauteur moyenne d'un arbre binaire choisi uniformément parmi les arbres binaires à n nœuds est en $\Theta(\sqrt{n})$.

Mais vu la méthode de construction des ABR, ils n'ont aucune raison de suivre la distribution uniforme sur les arbres binaires

Quelle distribution faut-il considérer ?

Observation : chaque arbre binaire à n sommets admet un unique étiquetage par $\{1, \ldots, n\}$ respectant les contraintes d'un ABR

Problème (?) :

Théorème (admis)

la hauteur moyenne d'un arbre binaire choisi uniformément parmi les arbres binaires à n nœuds est en $\Theta(\sqrt{n})$.

Mais vu la méthode de construction des ABR, ils n'ont aucune raison de suivre la distribution uniforme sur les arbres binaires

Quelle distribution faut-il considérer ?

Théorème

*la hauteur moyenne d'un ABR construit par l'insertion successive des entiers $1, \ldots, n$ dans un **ordre aléatoire choisi uniformément** est en $\Theta(\log n)$.*

C'est ce qu'on appelle la distibution de probabilité des ABR

On note :

- $\mathcal{B}_n$ l'ensemble des arbres binaires de taille n,
- $\mathcal{A}(\sigma)$ l'ABR obtenu par insertion successive de $\sigma(1), \sigma(2), \ldots$
- $h(a)$ la hauteur d'un arbre a.

On note :

- $\mathcal{B}_n$ l'ensemble des arbres binaires de taille n,
- $\mathcal{A}(\sigma)$ l'ABR obtenu par insertion successive de $\sigma(1), \sigma(2), \ldots$
- $h(a)$ la hauteur d'un arbre a.

On veut comparer deux notions de hauteur moyenne pour $\mathcal{B}_n$:

- selon la ***distribution uniforme :*** $\displaystyle\frac{1}{\#\mathcal{B}_n} \sum_{a \in \mathcal{B}_n} h(a)$
- et selon la ***distribution de probabilité des ABR :***

$$\frac{1}{\#\mathfrak{S}_n} \sum_{\sigma \in \mathfrak{S}_n} h(\mathcal{A}(\sigma)) = \frac{1}{\#\mathfrak{S}_n} \sum_{a \in \mathcal{B}_n} \#\{\sigma \in \mathfrak{S}_n \ \textit{tq}\ \mathcal{A}(\sigma) = a\} \cdot h(a)$$

On note :

- $\mathcal{B}_n$ l'ensemble des arbres binaires de taille n,
- $\mathcal{A}(\sigma)$ l'ABR obtenu par insertion successive de $\sigma(1), \sigma(2), \dots$
- $h(a)$ la hauteur d'un arbre a.

On veut comparer deux notions de hauteur moyenne pour $\mathcal{B}_n$:

- selon la ***distribution uniforme :*** $\displaystyle\frac{1}{\#\mathcal{B}_n} \sum_{a \in \mathcal{B}_n} h(a)$
- et selon la ***distribution de probabilité des ABR :***

$$\frac{1}{\#\mathfrak{S}_n} \sum_{\sigma \in \mathfrak{S}_n} h(\mathcal{A}(\sigma)) = \frac{1}{\#\mathfrak{S}_n} \sum_{a \in \mathcal{B}_n} \#\{\sigma \in \mathfrak{S}_n \;\; \textit{\textbf{tq}} \;\; \mathcal{A}(\sigma) = a\} \cdot h(a)$$

d'où l'importance de $\#\{\sigma \in \mathfrak{S}_n \text{ tq } \mathcal{A}(\sigma) = a\}$*, i.e. le nombre de permutations différentes telles que l'insertion successive de* $\sigma(1), \sigma(2), \dots$ *aboutisse à l'arbre* a

Combien de permutations produisent le même ABR ?

Cas filiforme : c'est le cas le plus simple :

Chacun des 2^{n-1} arbres filiformes de taille n compte seulement pour un poids 1 dans le calcul de la hauteur moyenne des ABR.

Combien de permutations produisent le même ABR ?

Cas filiforme : c'est le cas le plus simple :

Chacun des 2^{n-1} arbres filiformes de taille n compte seulement pour un poids 1 dans le calcul de la hauteur moyenne des ABR.

Plus généralement,

Lemme

Soit a un ABR, de racine r, et de sous-arbres g et d. Si $a = \mathcal{A}(\sigma)$, alors :

- *r a été inséré en premier :* $\sigma(1) = r$
- *soit γ l'ordre d'insertion des éléments de g (dans a) ; alors* $\mathcal{A}(\gamma) = g$
- *soit δ l'ordre d'insertion des éléments de d (dans a) ; alors* $\mathcal{A}(\delta) = d$
- *γ et δ s'intercalent de manière quelconque dans σ*

Combien de permutations produisent le même ABR ?

Cas filiforme : c'est le cas le plus simple :

> *Chacun des 2^{n-1} arbres filiformes de taille n compte seulement pour un poids 1 dans le calcul de la hauteur moyenne des ABR.*

Plus généralement,

Lemme

Soit a un ABR, de racine r, et de sous-arbres g et d. Si $a = \mathcal{A}(\sigma)$, alors :

- *r a été inséré en premier :* $\sigma(1) = r$
- *soit γ l'ordre d'insertion des éléments de g (dans a) ; alors* $\mathcal{A}(\gamma) = g$
- *soit δ l'ordre d'insertion des éléments de d (dans a) ; alors* $\mathcal{A}(\delta) = d$
- *γ et δ s'intercalent de manière quelconque dans σ*

Par exemple : pour $\sigma = 3\ 1\ 2\ 4$, on a $r = 3$, $\gamma = 1\ 2$ et $\delta = 4$.
Les seules permutations qui produisent le même ABR sont $3\ 1\ 4\ 2$ et $3\ 4\ 1\ 2$.

Combien de permutations produisent le même ABR ?

Le lemme permet un calcul récursif du nombre de permutations qui produisent un arbre donné :

1. calcul du nombre de possibilités pour g et pour d,
2. calcul du nombre de manières d'intercaler les éléments de g et ceux de d, *i.e.* colorier # g positions en vert et #d en rouge,
3. produit de ces 3 valeurs.

Combien de permutations produisent le même ABR ?

Le lemme permet un calcul récursif du nombre de permutations qui produisent un arbre donné :

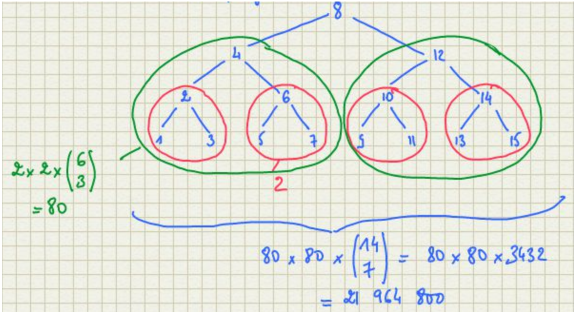
1. calcul du nombre de possibilités pour g et pour d,
2. calcul du nombre de manières d'intercaler les éléments de g et ceux de d, *i.e.* colorier # g positions en vert et #d en rouge,
3. produit de ces 3 valeurs.

Le mélange donne un facteur multiplicatif : $\binom{\#g + \#d}{\#g} = \frac{(\#g + \#d)!}{\#g!\,\#d!}$

(beaucoup) plus grand quand #g et #d sont du même ordre de grandeur que lorsque le partage est déséquilibré

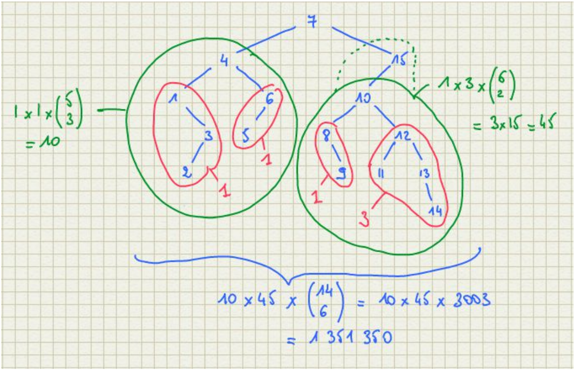
Combien de permutations produisent le même ABR ?

Exemple : cas de l'ABR parfait de taille 15



Combien de permutations produisent le même ABR ?

Un cas un peu moins équilibré



Combien de permutations produisent le même ABR ?

Moralité : la contribution des arbres est d'autant plus importante qu'ils sont équilibrés, et la différence d'ordre de grandeur est énorme entre les arbres relativement équilibrés et les arbres dégénérés comme les arbres filiformes.

Il est donc « moralement raisonnable » que la hauteur moyenne des ABR soit nettement plus faible que celle des arbres binaires uniformes... mais ce n'est pas une preuve !

Hauteur moyenne des ABR (démonstration)

$h(a)$ = hauteur de l'arbre a ; si g et d sont les sous-arbres de a :

$$h(a) = 1 + \max(h(g), h(d))$$

Hauteur moyenne des ABR (démonstration)

$h(a) =$ hauteur de l'arbre a ; si g et d sont les sous-arbres de a :

$$h(a) = 1 + \max(h(g), h(d))$$

seule majoration raisonnable : $h(a) \leqslant 1 + h(g) + h(d)$, *trop grossière pour espérer montrer mieux que* $h(a) \leqslant n$...

Hauteur moyenne des ABR (démonstration)

$h(a)$ = hauteur de l'arbre a ; si g et d sont les sous-arbres de a :

$$h(a) = 1 + \max(h(g), h(d))$$

seule majoration raisonnable : $h(a) \leqslant 1 + h(g) + h(d)$, *trop grossière pour espérer montrer mieux que* $h(a) \leqslant n$...

il faut introduire une grandeur plus sensible aux petites modifications :

Soit $H(a) = 2^{h(a)}$. On obtient alors une majoration plus fine :

$$H(a) = 2\max(H(g), H(d)) \leqslant 2(H(g) + H(d))$$

$H(a)$ *est en quelque sorte la « capacité » d'un arbre de même hauteur*

Hauteur moyenne des ABR (démonstration)

$h(a) =$ hauteur de l'arbre a ; si g et d sont les sous-arbres de a :

$$h(a) = 1 + \max(h(g), h(d))$$

seule majoration raisonnable : $h(a) \leqslant 1 + h(g) + h(d)$, *trop grossière pour espérer montrer mieux que* $h(a) \leqslant n$...

il faut introduire une grandeur plus sensible aux petites modifications :

Soit $H(a) = 2^{h(a)}$. On obtient alors une majoration plus fine :

$$H(a) = 2\max(H(g), H(d)) \leqslant 2(H(g) + H(d))$$

$H(a)$ *est en quelque sorte la « capacité » d'un arbre de même hauteur*

Soit $\overline{h}(n) = \frac{1}{n!} \sum_{\sigma \in \mathfrak{S}_n} h(\mathcal{A}(\sigma))$ et $\overline{H}(n) = \frac{1}{n!} \sum_{\sigma \in \mathfrak{S}_n} H(\mathcal{A}(\sigma))$

But : montrer que $\overline{h}(n) \in \Theta(\log n)$

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{H}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{H}(n)$

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{\mathrm{H}}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{\mathrm{H}}(n)$

$\Longrightarrow$ *il suffit de montrer que* $\overline{\mathrm{H}}(n)$ *est polynomiale (de degré quelconque)*

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{H}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{H}(n)$

$\Longrightarrow$ *il suffit de montrer que* $\overline{H}(n)$ *est polynomiale (de degré quelconque)*

on découpe $\mathfrak{S}_n$ en n « tranches » : $\mathfrak{S}_{n,i} = \{\sigma \in \mathfrak{S}_n \mid \sigma(1) = i\}$

$\mathfrak{S}_{n,i}$ est donc l'ensemble des σ t.q. $\mathcal{A}(\sigma)$ a i à la racine et donc

- un sous-ABR gauche g de taille $i - 1$ *(absolument quelconque)*
- un sous-ABR droit d de taille $n - i$ *(absolument quelconque)*

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{H}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{H}(n)$

$\Longrightarrow$ *il suffit de montrer que* $\overline{H}(n)$ *est polynomiale (de degré quelconque)*

on découpe $\mathfrak{S}_n$ en n « tranches » : $\mathfrak{S}_{n,i} = \{\sigma \in \mathfrak{S}_n \mid \sigma(1) = i\}$

$\mathfrak{S}_{n,i}$ est donc l'ensemble des σ t.q. $\mathcal{A}(\sigma)$ a i à la racine et donc

- un sous-ABR gauche g de taille $i-1$ *(absolument quelconque)*
- un sous-ABR droit d de taille $n-i$ *(absolument quelconque)*

Pour chaque arbre a, $H(a) \leqslant 2(H(g) + H(d))$, donc en moyennant sur $\mathfrak{S}_{n,i}$:

$$\overline{H}(n,i) \leqslant 2\left[\overline{H}(i-1) + \overline{H}(n-i)\right]$$

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{H}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{H}(n)$

$\Longrightarrow$ *il suffit de montrer que* $\overline{H}(n)$ *est polynomiale (de degré quelconque)*

on découpe $\mathfrak{S}_n$ en n « tranches » : $\mathfrak{S}_{n,i} = \{\sigma \in \mathfrak{S}_n \mid \sigma(1) = i\}$

$\mathfrak{S}_{n,i}$ est donc l'ensemble des σ t.q. $\mathcal{A}(\sigma)$ a i à la racine et donc

- un sous-ABR gauche g de taille $i-1$ *(absolument quelconque)*
- un sous-ABR droit d de taille $n-i$ *(absolument quelconque)*

Pour chaque arbre a, $H(a) \leqslant 2(H(g)+H(d))$, donc en moyennant sur $\mathfrak{S}_{n,i}$:

$$\overline{H}(n,i) \leqslant 2\left[\overline{H}(i-1) + \overline{H}(n-i)\right]$$

pour tout i, $\mathfrak{S}_{n,i}$ a le même cardinal $(n-1)!$, donc :

$$\overline{H}(n) = \frac{1}{n}\sum_{i=1}^{n} \overline{H}(n,i)$$

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{H}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{H}(n)$

$\Longrightarrow$ *il suffit de montrer que* $\overline{H}(n)$ *est polynomiale (de degré quelconque)*

on découpe $\mathfrak{S}_n$ en n « tranches » : $\mathfrak{S}_{n,i} = \{\sigma \in \mathfrak{S}_n \mid \sigma(1) = i\}$

$\mathfrak{S}_{n,i}$ est donc l'ensemble des σ t.q. $\mathcal{A}(\sigma)$ a i à la racine et donc

- un sous-ABR gauche g de taille $i-1$ *(absolument quelconque)*
- un sous-ABR droit d de taille $n-i$ *(absolument quelconque)*

Pour chaque arbre a, $H(a) \leqslant 2(H(g) + H(d))$, donc en moyennant sur $\mathfrak{S}_{n,i}$:

$$\overline{H}(n,i) \leqslant 2\left[\overline{H}(i-1) + \overline{H}(n-i)\right]$$

pour tout i, $\mathfrak{S}_{n,i}$ a le même cardinal $(n-1)!$, donc :

$$\overline{H}(n) \leqslant \frac{2}{n} \sum_{i=1}^{n} \left[\overline{H}(i-1) + \overline{H}(n-i)\right]$$

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{H}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{H}(n)$

$\Longrightarrow$ *il suffit de montrer que* $\overline{H}(n)$ *est polynomiale (de degré quelconque)*

on découpe $\mathfrak{S}_n$ en n « tranches » : $\mathfrak{S}_{n,i} = \{\sigma \in \mathfrak{S}_n \mid \sigma(1) = i\}$

$\mathfrak{S}_{n,i}$ est donc l'ensemble des σ t.q. $\mathcal{A}(\sigma)$ a i à la racine et donc

- un sous-ABR gauche g de taille $i-1$ *(absolument quelconque)*
- un sous-ABR droit d de taille $n-i$ *(absolument quelconque)*

Pour chaque arbre a, $H(a) \leqslant 2(H(g) + H(d))$, donc en moyennant sur $\mathfrak{S}_{n,i}$:

$$\overline{H}(n,i) \leqslant 2\left[\overline{H}(i-1) + \overline{H}(n-i)\right]$$

pour tout i, $\mathfrak{S}_{n,i}$ a le même cardinal $(n-1)!$, donc :

$$\overline{H}(n) \leqslant \frac{4}{n} \sum_{i=0}^{n-1} \overline{H}(i)$$

Hauteur moyenne d'un ABR (démonstration)

par convexité de l'exponentielle, $\overline{H}(n) \geqslant 2^{\overline{h}(n)}$, donc $\overline{h}(n) \leqslant \log \overline{H}(n)$

$\Longrightarrow$ *il suffit de montrer que* $\overline{H}(n)$ *est polynomiale (de degré quelconque)*

on découpe $\mathfrak{S}_n$ en n « tranches » : $\mathfrak{S}_{n,i} = \{\sigma \in \mathfrak{S}_n \mid \sigma(1) = i\}$

$\mathfrak{S}_{n,i}$ est donc l'ensemble des σ t.q. $\mathcal{A}(\sigma)$ a i à la racine et donc

- un sous-ABR gauche g de taille $i-1$ *(absolument quelconque)*
- un sous-ABR droit d de taille $n-i$ *(absolument quelconque)*

Pour chaque arbre a, $H(a) \leqslant 2(H(g) + H(d))$, donc en moyennant sur $\mathfrak{S}_{n,i}$:

$$\overline{H}(n,i) \leqslant 2\left[\overline{H}(i-1) + \overline{H}(n-i)\right]$$

pour tout i, $\mathfrak{S}_{n,i}$ a le même cardinal $(n-1)!$, donc :

$$\overline{H}(n) \leqslant \frac{4}{n} \sum_{i=0}^{n-1} \overline{H}(i)$$

et on vérifie par récurrence que $\frac{1}{4}\binom{n+3}{3}$ (polynôme de degré 3) est une majoration de $\overline{H}(n)$. $\square$

Corollaire

les opérations de recherche, d'insertion et de suppression ont une complexité (en temps) $\Theta(\log n)$ *(dans le pire cas de chaque ABR, en moyenne sur tous les ABR, considérés selon la distribution précédente)*

Corollaire

les opérations de recherche, d'insertion et de suppression ont une complexité (en temps) $\Theta(\log n)$ *(dans le pire cas de chaque ABR, en moyenne sur tous les ABR, considérés selon la distribution précédente)*

Corollaire

la construction d'un ABR par insertion successive de n *valeurs a une complexité* $\Theta(n \log n)$ ***en moyenne***

Corollaire

les opérations de recherche, d'insertion et de suppression ont une complexité (en temps) $\Theta(\log n)$ *(dans le pire cas de chaque ABR, en moyenne sur tous les ABR, considérés selon la distribution précédente)*

Corollaire

la construction d'un ABR par insertion successive de n *valeurs a une complexité* $\Theta(n \log n)$ ***en moyenne***

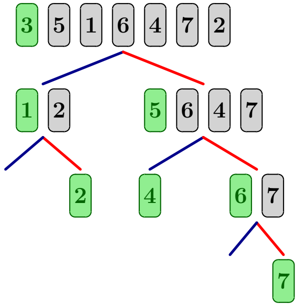
Corollaire

trier une liste par construction d'un ABR a une complexité $\Theta(n \log n)$ ***en moyenne***

Lemme

l'arbre de récursion de QuickSort, étiqueté par les pivots, est précisément l'ABR obtenu par insertion successive des pivots

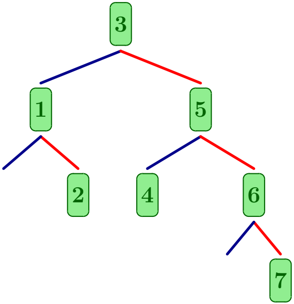
Exemple : `T = [3, 5, 1, 6, 4, 7, 2]`



Lemme

l'arbre de récursion de QuickSort, étiqueté par les pivots, est précisément l'ABR obtenu par insertion successive des pivots

Exemple : `T = [3, 5, 1, 6, 4, 7, 2]`



+ le cumul des complexités par niveau est en $O(n)$, donc :

Corollaire

la complexité de QuickSort *est également* $\Theta(n \log n)$ en moyenne

Mais..

Question : cette distribution est-elle réaliste ?

Mais..

Question : cette distribution est-elle réaliste ?

Demi-réponse en TP : vous devriez observer les mêmes formes de courbes tant que le nombre de suppressions reste raisonnable (quadratique) par rapport à la taille de l'arbre final

On peut donc compléter le tableau comparatif des complexités, avec une victoire nette des ABR sur les différents types de listes pour le traitement des ensembles dynamiques :

	tableau		liste chaînée		ABR
	non trié	trié	non triée	triée	
recherche	$\Theta(n)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(h)$
insertion	$+\Theta(1)$	$\Theta(n)$	$+\Theta(1)$	$+\Theta(1)$	$\Theta(h)$
suppression	$\Theta(n)$	$\Theta(n)$	$+\Theta(1)$	$+\Theta(1)$	$\Theta(h)$
minimum	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$	$\Theta(h)$
sélection	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(k)$	$\Theta(\log n)$
union	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n)$

On peut donc compléter le tableau comparatif des complexités, avec une victoire nette des ABR sur les différents types de listes pour le traitement des ensembles dynamiques :

	tableau		liste chaînée		ABR
	non trié	trié	non triée	triée	(en moyenne)
recherche	$\Theta(n)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(\log n)$
insertion	$+\Theta(1)$	$\Theta(n)$	$+\Theta(1)$	$+\Theta(1)$	$\Theta(\log n)$
suppression	$\Theta(n)$	$\Theta(n)$	$+\Theta(1)$	$+\Theta(1)$	$\Theta(\log n)$
minimum	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$	$\Theta(\log n)$
sélection	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(k)$	$\Theta(\log n)$
union	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n)$

On peut donc compléter le tableau comparatif des complexités, avec une victoire nette des ABR sur les différents types de listes pour le traitement des ensembles dynamiques :

	tableau		liste chaînée		ABR
	non trié	trié	non triée	triée	(en moyenne)
recherche	$\Theta(n)$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(\log n)$
insertion	$+\Theta(1)$	$\Theta(n)$	$+\Theta(1)$	$+\Theta(1)$	$\Theta(\log n)$
suppression	$\Theta(n)$	$\Theta(n)$	$+\Theta(1)$	$+\Theta(1)$	$\Theta(\log n)$
minimum	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$	$\Theta(\log n)$
sélection	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(k)$	$\Theta(\log n)$
union	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n)$

(pour l'union : en commençant par un (des) parcours infixes ; ou sans construire de listes auxiliaires, mais c'est plus compliqué)
(pour la sélection : enrichir la structure, par exemple en stockant dans chaque nœud la taille du sous-arbre correspondant)

Comment assurer un coût logarithmique ?

i.e. contraindre les ABR à rester « raisonnablement » équilibrés

- les arbres rouges-noirs
- les arbres AVL (dus à Adelson-Velskii et Landis)
- ...

Comment assurer un coût logarithmique ?

i.e. contraindre les ABR à rester « raisonnablement » équilibrés

- les arbres rouges-noirs

  - ABR complétés (par des feuilles noires)
  - noeuds rouges ou noirs
  - le père d'un sommet rouge est noir
  - la racine est noire
  - chaque branche a le même nombre de sommets noirs

- les arbres AVL (dus à Adelson-Velskii et Landis)
- ...

Comment assurer un coût logarithmique ?

i.e. contraindre les ABR à rester « raisonnablement » équilibrés

- les arbres rouges-noirs

  > - ABR complétés (par des feuilles noires)
  > - noeuds rouges ou noirs
  > - le père d'un sommet rouge est noir
  > - la racine est noire
  > - chaque branche a le même nombre de sommets noirs

- les arbres AVL (dus à Adelson-Velskii et Landis)

  > pour chaque nœud, les hauteurs des deux sous-arbres diffèrent au plus de 1

- …

Comment assurer un coût logarithmique ?

i.e. contraindre les ABR à rester « raisonnablement » équilibrés

- les arbres rouges-noirs
- les arbres AVL (dus à Adelson-Velskii et Landis)
- ...

Théorème

- *la hauteur d'un arbre rouge-noir à n sommets est au plus* $2 \log n$
- *la hauteur d'un AVL à n sommets est au plus* $1.44 \log n$

Comment assurer un coût logarithmique ?

i.e. contraindre les ABR à rester « raisonnablement » équilibrés

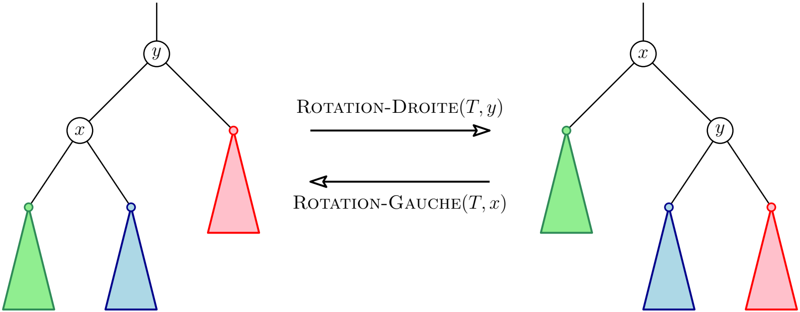
- les arbres rouges-noirs
- les arbres AVL (dus à Adelson-Velskii et Landis)
- ...

Théorème

- *la hauteur d'un arbre rouge-noir à* n *sommets est au plus* $2 \log n$
- *la hauteur d'un AVL à* n *sommets est au plus* $1.44 \log n$

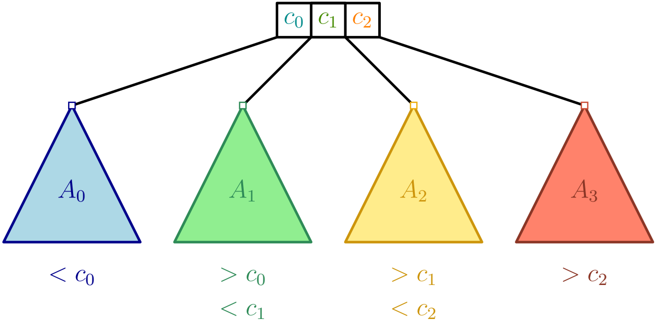
Inconvénient : les opérations d'insertion et de suppression sont (nettement) plus complexes

Outils de rééquilibrage : les rotations

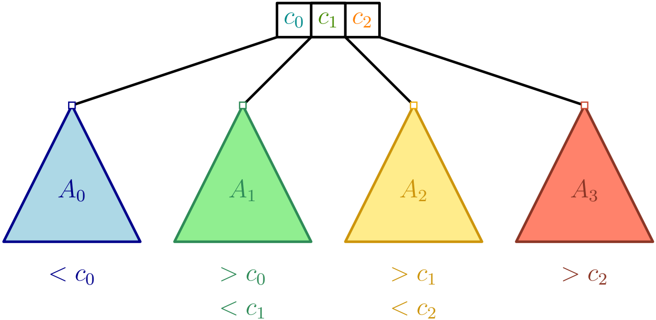


- imposer que toutes les feuilles soient au même niveau
- en contrepartie, lever (en partie) la contrainte d'arité : un nœud peut avoir plus de 2 fils

- imposer que toutes les feuilles soient au même niveau
- en contrepartie, lever (en partie) la contrainte d'arité : un nœud peut avoir plus de 2 fils



- imposer que toutes les feuilles soient au même niveau
- en contrepartie, lever (en partie) la contrainte d'arité : un nœud peut avoir plus de 2 fils



k clés $\iff$ k + 1 fils

B-arbres d'ordre p $(p \geqslant 1)$:

- chaque nœud ou feuille contient au plus $2p$ clés ;
- chaque nœud ou feuille (sauf la racine) contient au moins p clés ;
- un nœud d'arité $k+1$ contient exactement k clés ;
- toutes les feuilles ont la même profondeur
- si un nœud contient les clés $c_0 < c_1 < \cdots < c_{k-1}$ et possède les sous-arbres A_0, A_1... A_k, tous les éléments de A_i sont supérieurs à c_{i-1} (si $i > 0$) et inférieurs à c_i (si $i < k$).

B-arbres d'ordre p $(p \geqslant 1)$:

- chaque nœud ou feuille contient au plus $2p$ clés ;
- chaque nœud ou feuille (sauf la racine) contient au moins p clés ;
- un nœud d'arité $k+1$ contient exactement k clés ;
- toutes les feuilles ont la même profondeur
- si un nœud contient les clés $c_0 < c_1 < \cdots < c_{k-1}$ et possède les sous-arbres A_0, A_1... A_k, tous les éléments de A_i sont supérieurs à c_{i-1} (si $i > 0$) et inférieurs à c_i (si $i < k$).

(définition fluctuante... Cormen par exemple choisit d'avoir entre p et $2p$ *fils*)

Lemme

le parcours infixe fournit la liste triée des clés

Lemme

le parcours infixe fournit la liste triée des clés

Lemme

la hauteur d'un B-arbre d'ordre p *contenant* n *clés est comprise (à peu près...) entre* $\log_{2p+1} n$ *et* $\log_{p+1} n$

Lemme

le parcours infixe fournit la liste triée des clés

Lemme

la hauteur d'un B-arbre d'ordre p *contenant* n *clés est comprise (à peu près...) entre* $\log_{2p+1} n$ *et* $\log_{p+1} n$

Lemme

la recherche d'un élément a complexité $\Theta(\log n)$ *au pire*

Lemme

le parcours infixe fournit la liste triée des clés

Lemme

la hauteur d'un B-arbre d'ordre p *contenant* n *clés est comprise (à peu près...) entre* $\log_{2p+1} n$ *et* $\log_{p+1} n$

Lemme

la recherche d'un élément a complexité $\Theta(\log n)$ *au pire*

Lemme

l'insertion et la suppression d'une clé a complexité $\Theta(\log n)$ *au pire*