

TD 2, Analyse amortie

Petites relations utiles :

- $\sum_{i=1}^n i = 1 + 2 + \dots + n = \frac{n(n+1)}{2}$,
- $\sum_{i=1}^n 2^i = 2^0 + 2^1 + \dots + 2^n = 2^{n+1} - 1$,
- $2^k = n$ si et seulement si $k = \log_2 n$
- pour tout $k \geq 0$, $n + \frac{n}{2} + \frac{n}{2^2} + \dots + \frac{n}{2^k} = n \left(1 + \frac{1}{2} + \frac{1}{2^2} + \dots + \frac{1}{2^k}\right) \leq 2n$.

Exercice 1. Analyse amortie pour des structures à croissance exponentielle

Une suite de m opérations est effectuée sur une structure de données. La i -ème opération coûte i si i est une puissance exacte de 2, et 1 sinon. Déterminer le coût amorti de l'opération par la méthode de l'agréat. On pourra supposer que $m = 2^k$.

Exercice 2. Méthode comptable sur l'exemple de la pile du cours: ajout de PILEVIDE

Dans la méthode comptable du cours, on avait supposé un peu abusivement que PILEVIDE ne coûtait rien. Supposez maintenant les coûts suivants et trouvez les bons coûts amortis (s est le nombre d'éléments dans la pile à ce moment de l'exécution).

Opération	coût réel	coût amorti
EMPILER(S, x)	1	?
DÉPILER(S)	1	?
MULTIDÉPILER(S, k)	$\min(s, k)$?
PILEVIDE(S)	1	?

Exercice 3. Méthode comptable sur l'exemple de la pile du cours: ajout des coûts réels

On affecte les coûts réels suivants aux fonctions de pile.

Opération	coût réel	coût amorti
EMPILER(S, x)	a	?
DÉPILER(S)	b	?
MULTIDÉPILER(S, k)	$c + d \min(s, k)$?
PILEVIDE(S)	e	?

Que faut-il choisir comme coûts amortis pour que le raisonnement précédent continue à fonctionner ?

Exercice 4. Méthode comptable sur l'exemple du compteur du cours: ajout de RÉINITIALISER

Imaginons qu'on rajoute une méthode RÉINITIALISER qui remet le compteur à 0. Comment faire évoluer la structure de compteur pour qu'une séquence quelconque de n opérations INCRÉMENTER ou RÉINITIALISER prenne un temps $\mathcal{O}(n)$. On supposera le compteur initialement à 0. Donnez les coûts réels et amortis des deux opérations.

Exercice 5. Pile avec sauvegarde régulière

Soit une pile avec les 3 fonctions usuelles (EMPILER(S, x), DÉPILER(S), PILEVIDE(S)). On rajoute une fonctionnalité à la pile qui est que toutes les k opérations, on fait une copie de sauvegarde de toute la pile. On suppose de plus que la pile ne dépasse jamais k valeurs. Montrer alors, en choisissant bien les coûts amortis, que le coût amorti total de m opérations (sauvegardes incluses) est bien $\mathcal{O}(m)$.

Exercice 6. Analyse amortie d'une file implémentée à l'aide de deux piles

Montrer que l'on peut implémenter une file avec deux piles ordinaires, de telle manière que le coût amorti de chaque opération ENFILER et DÉFILER soit $O(1)$.

Ecrivez d'abord les pseudo-codes de la procédure ENFILER(F, x) et de la fonction DEFILER(F). On pourra supposer que $F.P1$ et $F.P2$ sont les deux piles associées à F .

Utilisez ensuite la méthode de l'agrégat, puis la méthode comptable pour trouver les coûts amortis.

Exercice 7. Analyse amortie dans un ABR

Soit A un arbre binaire de recherche, contenant n éléments et de profondeur maximum k . Soient les fonctions:

- **Fonction** Premier($\underline{E} A : \text{ABR}$) : Noeud qui retourne le Noeud de l'arbre dont la valeur est la plus faible.
- **Fonction** Suivant($\underline{E} A : \text{ABR}, \underline{E} N : \text{Noeud}$) : Noeud qui retourne le Noeud de l'arbre dont la valeur est juste supérieure à la valeur du noeud N ou Null sinon.

1. Ecrire le pseudo-code des fonctions *Premier* et *Suivant*, en utilisant les fonctions classiques des ABR: *Racine*, *Gauche*, *Droite*, *Pere*.
2. Soit le programme suivant. Que fait-il ? Quelle est la complexité en pire cas de l'opération *Suivant* ? En déduire une borne supérieure sur la complexité en pire cas du programme *Parcours*.

Action Parcours($\underline{E} A : \text{Arbre}$) ;

Var : $N : \text{Noeud}$;

début

$N \leftarrow \text{Premier}(A)$;

Tant Que $N \neq \text{Null}$ **Faire**

 Affiche(Valeur(A, N)) ;

$N \leftarrow \text{Suivant}(A, N)$;

3. Quelle est maintenant le coût amorti de chaque appel à *Suivant* ? Utilisez la méthode des agrégats.

Exercice 8. Elimination des $n/2$ plus grands éléments en temps amorti constant

Imaginer une structure de données qui permettent les opérations suivantes pour un ensemble S d'entiers:

- INSÉRER(S, x) insère x dans l'ensemble S .
- SUPPRIMEMOITIÉSUP(S) supprime les $\lceil \frac{\text{Card}(S)}{2} \rceil$ plus grands éléments de S .

Comment implémenter ces fonctions pour que toute séquence de m opérations soit exécutée en temps $\mathcal{O}(m)$. Utilisez la méthode de votre choix.

Note: On rappelle qu'il existe un algorithme en temps linéaire pour déterminer la médiane dans un tableau de n éléments.