TD Algo2 – session 1 – Tas

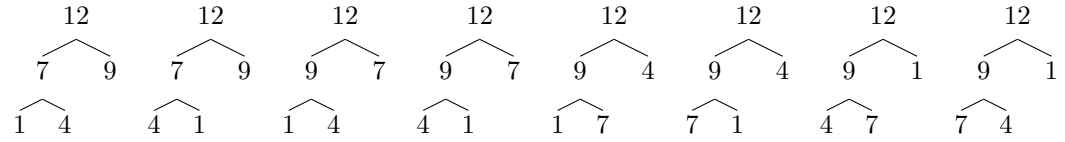
12 mars 2025

Objectifs d'apprentissage :
— manipuler la structure de tas et étudier les implications de la propriété de tas ;
— concevoir des algorithmes relatifs aux tas ;
— mettre en pratique des techniques de preuve pour la validité ou la complexité d'algorithmes.

> Les 9 premiers exercices sont essentiels. Plus d'exercices sur `https://algs4.cs.princeton.edu/24pq/index.php`.

Exercice 1 : Énumération

Dessiner tous les tas max possibles avec les éléments suivants : 1, 4, 7, 9 et 12.



Exercice 2 : Position

Dans un tas max où tous les éléments sont distincts, quelles sont les positions possibles pour le plus petit élément ?

> N'importe quelle feuille.

Exercice 3 : Hauteur

Quels sont les nombres minimal et maximal d'éléments dans un tas de hauteur h ?

> Arbre binaire complet de hauteur h (nombre maximal) : $2^{h+1} - 1$. Avec un nœud supplémentaire (donc 2^{h+1}), on atteint la hauteur $h + 1$. Donc le nombre minimal pour une hauteur de h est 2^h.

Exercice 4 : Tableau trié

Un tableau trié dans l'ordre décroissant forme-t-il un tas max ? Proposer un argument par récurrence.

> La propriété de tas max est directement vérifiée pour chaque indice car le parent de chaque nœud a un indice inférieure et donc une valeur supérieure dans un tableau trié dans l'ordre décroissant : $A[i/2] \geq A[i]$ pour $1 \leq i \leq n$.

Exercice 5 : Vérification

Concevoir un algorithme de certification en temps linéaire pour vérifier si un tableau est un tas max.

```
VÉRIFIER-TAS-MAX(A)
pour i = A.n decr jusqu'à 2
  si A[⌊i/2⌋] < A[i] alors
    retourner FAUX
retourner VRAI
```

Exercice 6 : Entasser tas min

En s'inspirant de la procédure ENTASSER-MAX, écrire le pseudo-code pour la procédure ENTASSER-MIN(A, i) qui fait la même chose mais pour un tas min.

Écrire la procédure ENTASSER-MAX découle directement du cours. Il y a 2 comparaisons à inverser.

```
ENTASSER-MIN(A, i)
g ← GAUCHE(i)
d ← DROITE(i)
si g ≤ A.n et A[g] < A[i] alors
  min ← g
sinon min ← i
si d ≤ A.n et A[d] < A[min] alors
  min ← d
si min ≠ i alors
  échanger A[i] et A[min]
  ENTASSER-MIN(A, min)
```

Exercice 7 : Invariant

Quel invariant de boucle permettrait de prouver la validité de l'algorithme du tri par tas ?

Au début de chaque itération de la boucle, le sous-tableau $A[1 \dots i]$ est un tas max contenant les i plus petits éléments de $A[l \dots n]$ et le sous-tableau $A[i+1 \dots n]$ contient les $n-i$ plus grands éléments de $A[1 \dots n]$ triés.

La preuve se ferait via les 3 étapes suivantes. L'initialisation se vérifie pour $i = n$ car $A[1 \dots n]$ est un tas qui vient d'être construit par l'instruction qui précède la boucle.

Pour la conservation, on suppose que l'invariant est vrai pour un i donné. Comme on extrait l'élément maximal de $A[1 \dots i]$ et qu'il est plus petit que les éléments de $A[i+1 \dots n]$, $A[i \dots n]$ contient bien les $n-i+1$ plus grand éléments de $A[1 \dots n]$ triés. L'entassement rétablie bien la propriété de tas max sur $A[1 \dots i-1]$ car les sous-arbre enracinés en 2 et 3 était bien des tas max.

La terminaison signifie que lorsque $i = 1$, $A[2 \dots n]$ contient les $n-1$ plus grands éléments triés. Comme $A[1]$ est le plus petit élément, cela conclut la preuve.

Exercice 8 : Choix d'une structure

Supposons qu'une application comporte un grand nombre d'opérations d'insertion, mais seulement quelques opérations de suppression du maximum. Quelle implémentation vous semble la plus efficace : le tas, le tableau non ordonné, le tableau ordonné ?

Il faut la structure la plus efficace pour l'insertion : c'est le tableau non ordonné.

Exercice 9 : Complexité de la construction d'un tas

Quel est le plus petit nombre de comparaisons et d'échanges pour construire un tas à partir d'un tableau de n éléments ?

Entasser un tableau de n éléments dans l'ordre décroissant nécessite 0 échange et $n-1$ comparaisons.
À l'inverse, entasser un tableau de n éléments dans l'ordre croissant nécessite environ n échanges et $2n$ comparaisons car chaque échange entraîne une comparaison avec les 2 enfants.

Exercice 10 : Complexité du tri par tas

Quel est le nombre de comparaisons pour trier un tableau de longueur n avec le tri par tas dans le meilleur et pire cas ?

Si on autorise les éléments identiques, le meilleur cas est linéaire. Sinon, le meilleur cas est d'environ $n \log n$ comparaisons (avec une entrée non triviale). Le pire cas est environ $2n \log n$ comparaisons.

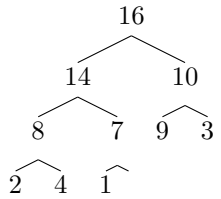
Exercice 11 : Complexité pour des données triées

Quel est le temps d'exécution du tri par tas sur un tableau A de longueur n déjà trié en ordre croissant ? Et en ordre décroissant ?

En ordre croissant, tout les éléments changent de place pour créer le tas max. C'est donc du $O(n \log n)$. En ordre décroissant, la création du tas max de modifie pas le tableau mais la seconde partie du tri si. C'est aussi du $O(n \log n)$.

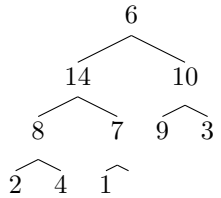
Exercice 12 : Représentation

Donner la représentation en tableau du tas suivant :



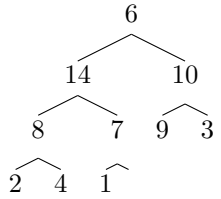
Exercice 13 : Entasser

Entasser l'élément 6 dans le tas suivant :



Exercice 14 : Supprimer

Supprimer le sommet du tas suivant :



Exercice 15 : Construction

Illustrer l'action de CONSTRUIRE-TAS-MAX sur le tableau $A = (5, 3, 17, 10, 84, 19, 6, 22, 9)$.

Exercice 16 : Tri

Illustrer l'action de TRI-PAR-TAS sur le tableau $A = (5, 13, 2, 25, 7, 17, 20, 8, 4)$.