Analyse d'algorithmes et programmation - 1^{er} contôle continu

Enseignants: Christina Boura, Gaëtan Leurent

2 mars 2017

Durée du contrôle : 1h30.

Les documents ainsi que l'accès aux programmes élaborés pendant les séances de TP sont autorisés.

Bonne chance !

1 Questions

Exercice 1

(a) Démontrer que $n! = \mathcal{O}(n^n)$ et que $\log(n!) = \mathcal{O}(n \log(n))$.

Démonstration : On a

$$n! = 1 \cdot 2 \cdots n \leq n \cdot n \cdots n = n^n, \text{ pour tout } n \geq 1.$$

(b) Démontrer que $1 + 2 + \cdots + n = \Theta(n^2)$ ou donner un contre-exemple.

Démonstration : D'un côté on a

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2} = \frac{n^2 + n}{2} \leq n^2, \text{ pour tout } n \geq 1,$$

car $n \leq n^2$ pour $n \geq 1$. De l'autre côté,

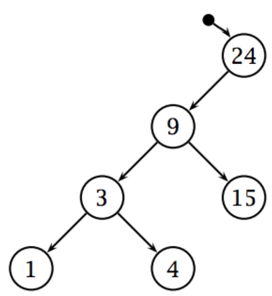
$$\frac{n(n+1)}{2} \geq \frac{n^2}{2} \text{ pour tout } n \geq 1.$$

Par conséquent,

$$\frac{1}{2}n^2 \leq 1 + 2 + \cdots + n \leq n^2, \text{ pour tout } n \geq 1$$

et ceci prouve que $1 + 2 + \cdots + n = \Theta(n^2)$.

Exercice 2 On considère l'arbre binaire de recherche ci-dessous :



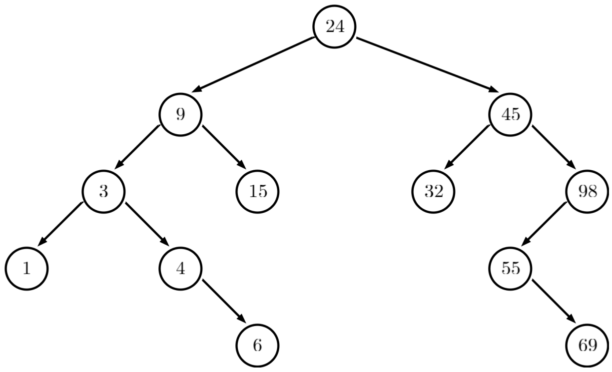
(a) Donner 5 ordres d'insertion possibles (permutations) des clés qui auraient pu produire cet arbre.

Correction : Ci-dessous sont 5 ordres d'insertion possibles (mais il y en a d'autres) :

1. $[24, 9, 15, 3, 4, 1]$
2. $[24, 9, 15, 3, 1, 4]$
3. $[24, 9, 3, 15, 4, 1]$
4. $[24, 9, 3, 15, 1, 4]$
5. $[24, 9, 3, 4, 15, 1]$

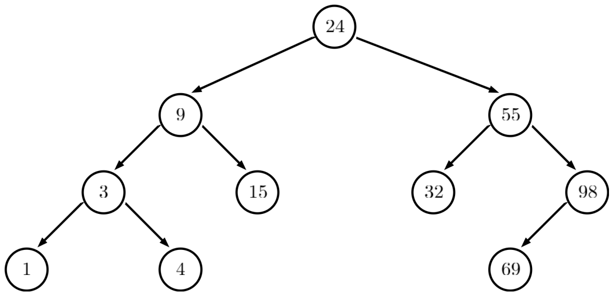
(b) Dessiner ce même arbre après l'insertion des clés 6, 45, 32, 98, 55 et 69 en respectant cet ordre.

Correction :

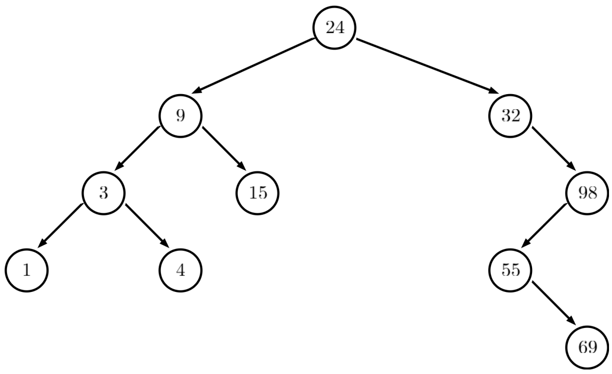


(c) Dessiner l'arbre résultant de la question (b) après la suppression des clés 6 et 45. Quelle est sa hauteur ?

Correction :



La hauteur de cet arbre est $h = 3$. Alternativement, on peut aussi effectuer la suppression de la façon suivante :



La hauteur de cet arbre est $h = 4$.

(d) Dessiner un arbre binaire de recherche contenant les mêmes clés que l'arbre de la question (c) et ayant une hauteur $h = 3$.

Correction : Le premier arbre de la question (c) répond à la question.

Exercice 3 Le temps d'execution $T(n)$ d'un algorithme de tri est donné en fonction du nombre n d'éléments à trier par la formule récursive suivante :

$$T(n) = \begin{cases} 0 & \text{si } n = 1, \\ 3T(n/3) + 2cn & \text{sinon,} \end{cases}$$

avec c une constante positive. Résoudre cette équation afin de dériver une formule explicite pour $T(n)$ si on suppose que $n = 3^m$. Donner la complexité de cet algorithme en utilisant la notation $\mathcal{O}$.

Correction :

$$
\begin{aligned}
T(3^m) &= 3T(3^{m-1}) + 2c3^m \\
&= 3[3T(3^{m-2}) + 2c3^{m-1}] + 2c3^m \\
&= 3^2T(3^{m-2}) + 4c3^m \\
&= 3^2[3T(3^{m-3}) + 2c3^{m-2}] + 4c3^m \\
&= 3^3T(3^{m-3}) + 6c3^m \\
&= \vdots \\
&= 3^mT(1) + 2c \cdot m3^m \\
&= 2c \cdot m3^m \\
&= 2c \cdot n \log_3(n).
\end{aligned}
$$

On conclue alors que $T(n) = \mathcal{O}(n \log_3(n))$.

Exercice 4 On s'intéresse au nombre minimal de produits scalaires nécessaires afin de calculer le produit matriciel $M_1M_2M_3M_4M_5$ où la dimension de la matrice M_i est notée $p_{i-1} \times p_i$ (p_{i-1} lignes et p_i colonnes) pour $1 \leq i \leq 5$. Si on note $c_{i,j}$, $j \geq i$, le nombre minimal de produits scalaires nécessaires pour calculer le produit $M_iM_{i+1} \cdots M_j$, donner l'expression de $c_{2,5}$ en fonction des quantités $c_{i,j}$ avec $j - i < 3$ et des variables p_i.

Correction :

$$c_{2,5} = \min\{c_{2,2} + c_{3,5} + p_1p_2p_5, c_{2,3} + c_{4,5} + p_1p_3p_5, c_{2,4} + c_{5,5} + p_1p_4p_5\}.$$