TP Algorithmique

Lucas Brifault

9 juin 2017

Arbre Binaire

Question 2

On s'intéresse à la propriété $\mathcal{P}(n)$: " un arbre binaire complet à n feuilles possède $2n-1$ noeuds." définie pour $n \in \mathbb{N}^*$.

— Si $n = 1$, on a une feuille dans l'arbre $\mathcal{A}_1$, si cette feuille avait un parent, alors ce noeud parent aurait un autre enfant. comme l'arbre est fini, on trouverait au moins deux feuilles dans la branche de cet autre enfant, ce qui contredit le fait que $n = 1$ d'où la feuille de départ n'a pas de parent, c'est le seul noeud de l'arbre. On a donc bien $2n - 1 = 1$ noeud et $\mathcal{P}(1)$ est vraie.

— supposons qu'il existe $n \in \mathbb{N}^*$ tel que $\mathcal{P}(n)$ est vraie. on considère alors un arbre binaire complet à $n+1$ feuilles, $\mathcal{A}_{n+1}$. soit f une feuille de cet arbre. Elle a forcément un parent cfar sinon on se retrouve dans le cas précédent, et c'est la seule feuille de l'arbre, or $n + 1 \geq 2$. ce parent p a un autre enfant f'. Si on considère l'arbre $\mathcal{A}_n$ que l'on définit comme l'arbre $\mathcal{A}_{n+1}$ sans f et f', on a bien un arbre complet à n feuilles (où p est devenu une feuille). on peut donc lui appliquer la peopriété $\mathcal{P}(n)$: il possède $2n - 1$ noeuds. Comme $\mathcal{A}_{n+1}$ possède 2 noeuds de plus que $\mathcal{A}_n$, (f et f'), on a que $\mathcal{A}_{n+1}$ a bien, $2n - 1 + 2 = 2(n + 1) - 1$ noeuds. d'où $\mathcal{P}(n + 1)$ vraie.

Question 3

Il suffit de considérer un arbre à deux noeuds, un parent p et son enfant f, qui est donc une feuille. On a $n = 1$ et pourtant le nombre de noeuds est $2 \neq 1 = 2n - 1$.

Codage de Huffman

Question 6

Supposons qu'un caractère c soit codé par $s_1...s_d$ et qu'un autre caractère c' soit codé par $s'_1...s'_r$ avec $r \geq d$ et tel que $s'_i = s_i$ pour tout $i \in \{1, ..., d\}$. $s_1 = s'_1$ signifie que les deux feuilles associées aux caractères c et c' descendent du même enfant N_1 de la racine N_0. De même, $s_2 = s'_2$ veut dire que ces deux feuilles descendent également du même enfant N_2 de N_1, et ainsi de suite jusqu'au fait que le feuille reliée c' est dans la même branche que la feuille reliée à c par rapport au parent de c, ce qui signifie que soit c'est la même feuille, $d = r$ et $c = c'$, soit la feuille de c' descend du noeud associé à c, ce qui contredit la construction de l'arbre puisque les caractères sont uniquement associés à des feuilles. d'où le code d'un caractère ne peut pas être le préfixe d'un caractère différent dans le code de Huffman.
C'est particulièrement utile pour une question d'efficacité et de compression, on a pas besoin de séparer les codes des caractères, on peut tout-à-fait les concaténer les uns à la suite des autres, le programme qui recevra le message codé pourra sans problème le décoder en temps réel (et pour chaque séquence qui correspond à un caractère, il n'aura pas à vérifier que cette séquence ne participe pas à coder le début d'un autre caractère).

Question 7

L'équilibre dépend des différences de fréquence entre les caractères. Supposons par exemple que l'on code 5 caractères, $c_1, ..., c_5$ (feuilles $f_1, ..., f_5$) tels que les fréquences associées $\nu_1, ..., \nu_5$ soient réparties de la sorte :

$$\begin{aligned} \nu_1 &= 0.6 \\ \nu_2 &= ... = \nu_5 = 0.1 \end{aligned}$$

On va alors fusionner les feuilles f_2 et f_3 d'une part (en un noeud p_{23} de fréquence 0.2) et les feuilles f_4 et f_5 d'autre part (en un noeud p_{45} de fréquence 0.2). Ensuite on va fusionner les noeuds p_{23} et p_{45}, de fréquence plus faible que f_1, pour former p_{2345}, de fréquence 0.4. Finalement on fusionnera f_1 et p_{2345} pour obtenir l'arbre final, où f_1 sera de profondeur 1 et $f_2, ..., f_5$ de profondeur 3.

Question 9

Il nous faut utiliser une file de priorité, en effet, l'algorithme consiste à traiter en priorité les éléments (noeuds) qui ont la fréquence d'apparition la plus faible, il faut donc pouvoir trouver efficacement ces noeuds-ci et y accéder, sans avoir à re-trier la liste à chaque fois.

0.1 Question 10

Ici j'ai utilisé une classe "Couple" plutôt que "pair".