Travaux dirigés

Théorème maître

On considère ici que $c(n)$ est le nombre d'opérations nécessaires à un algorithme sur une entrée de taille n, et on suppose qu'il existe sur la suite $(c(n))_n$ une relation de récurrence de la forme $c(n) = ac(\lceil n/2 \rceil) + bc(\lfloor n/2 \rfloor) + d(n)$, avec $a + b \geq 1$.

1. Écrire une fonction `complexite` qui prend en paramètres a, b, d et n et renvoie $c(n)$ (on considèrera *sans perte de généralité* que $c(0) = 1$)
2. Donner deux exemples d'algorithmes dont la complexité satisfait une telle relation.
3. Montrer que si d est croissante, c l'est aussi.
4. Étudier la valeur de $c(n)$ pour n une puissance de 2. *On supposera que d est polynomial en n*
5. Déduire de ce qui précède le **théorème maître** : lorsque $a + b > 1$, que la suite $(d_n)_n$ est croissante, avec $d(n) = \Theta(n^k)$,
   - si $\log(a + b) < k, c_n = \Theta(n^k)$
   - si $\log(a + b) = k, c_n = \Theta(n^k \log(n))$
   - si $\log(a + b) > k, c_n = \Theta(n^{\log(a+b)}$
6. Comparer le résultat théorique à des cas particuliers à l'aide de `complexite`.
7. Comment faire si un algorithme divise un problème en trois sous-problèmes de taille équivalente au lieu de deux ? Le résultat est-il modifié ? Dans quel sens ?