EA3

Complexité en temps de l'algorithme de dichotomie

Etant donné un tableau T et un élément x, si x n'appartient pas à T, alors l'algorithme de dichotomie fait des appels récursifs jusqu'à ce que le tableau passé en paramètre de l'appel récursif soit vide (lorsque $deb > fin$). Par contre, si x appartient à T, l'algorithme peut s'arrêter plus tôt et donc, faire moins d'appels récursifs. Donc le nombre d'appels récursifs est au pire le nombre d'appels faits lorsque x n'appartient à T.
Par ailleurs, lors de chaque appel récursif, il y a au plus 2 comparaisons et une affectation de faites. Donc la complexité, repose sur le nombre d'appels récursifs.
Soit n la taille de T, c'est-à-dire que T a n éléments. A chaque appel récursif, la taille du tableau est divisée par 2. Au premier appel récursif, la taille du tableau est $\frac{n}{2}$. Au deuxième appel récursif, la taille du tableau est $\frac{n}{2^2}$... Au i-ième appel récursif, la taille du tableau est $\frac{n}{2^i}$.
Soit k l'entier tel que $2^{k-1} \leqslant n < 2^k$. Alors $0 \leqslant \frac{n}{2^k} < 1$. Donc au k-ième appel récursif, la taille du tableau est strictement inférieure à 1, donc le tableau est vide et la récursion s'arrête. Au pire, il y aura donc k appels récursifs. Or, $2^{k-1} \leqslant n < 2^k \Leftrightarrow k - 1 \leqslant log_2(n) < k$. Donc au pire, il y aura de l'ordre de $log_2(n)$ appels récursifs.
La complexité au pire de l'algorithme de dichotomie en nombre de comparaisons, ou en nombre d'affectations, est donc de l'ordre de $log_2(n)$.