Corrigé du Devoir Libre n° 10*

Une preuve du Théorème Maître [1]

A. Rappels et notations

B. Problématique

C. Questions préliminaires

1. Soit $x>0$. $\boxed{b^{\log_b(x)}=\exp(\log_b(x)\ln b)=\exp(\ln x)=x.}$

2. Par récurrence forte sur $n\in\mathbb{N}$, on pose $P(n)$: « $T(n)>0$ ».
Par hypothèse initiale, on a déjà $T(0)>0$ et $T(1)>0$.
Soit $n\in\mathbb{N}^*$ tel que pour tout $k\in[\![0,n-1]\!]$, $T(k)>0$.
Alors $T(n)=\alpha T\left(\left\lfloor\frac{n}{b}\right\rfloor\right)+\beta T\left(\left\lceil\frac{n}{b}\right\rceil\right)+u_n>0$ par hypothèse de récurrence (on a bien $\left\lfloor\frac{n}{b}\right\rfloor<n$ et $\left\lceil\frac{n}{b}\right\rceil<n$) et hypothèses sur α, β et u_n, ce qui établit la récurrence.
Ainsi, $\boxed{\text{pour tout } n\in\mathbb{N},\ T(n)>0.}$

D. Majoration de $T(n)$

1. On a, par définition, $\log_b(n)\leqslant\ell(n)<\log_b(n)+1$, donc $1\leqslant\dfrac{\ell(n)}{\log_b(n)}<1+\dfrac{\ln b}{\ln n}$ donc, par encadrement, $\boxed{\dfrac{\ell(n)}{\log_b(n)}\xrightarrow[n\to+\infty]{}1.}$

2. Soit $n>0$. $\left\lceil\frac{n}{b}\right\rceil<\frac{n}{b}+1$ donc $b\left\lceil\frac{n}{b}\right\rceil<n+b$ puis $b\left\lceil\frac{n}{b}\right\rceil\leqslant n+b-1$ car tout est entier, donc $\boxed{\left\lceil\frac{n}{b}\right\rceil\leqslant 1+\frac{n-1}{b}.}$

3. Soit $n\geqslant b$. Par définition,
$$\left\lceil\frac{n}{b}\right\rceil-1<\frac{n}{b}\leqslant\left\lceil\frac{n}{b}\right\rceil$$
donc, par croissance de $\log_n$,
$$\log_b\left(\left\lceil\frac{n}{b}\right\rceil-1\right)<\log_b\left(\frac{n}{b}\right)=\log_b n-1\leqslant\log_b\left(\left\lceil\frac{n}{b}\right\rceil\right)\leqslant\left\lceil\log_b\left(\left\lceil\frac{n}{b}\right\rceil\right)\right\rceil=\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)\in\mathbb{N}.$$
Remarquons ensuite qu'il n'y a pas d'entier dans l'intervalle $\left]\log_b\left(\left\lceil\frac{n}{b}\right\rceil-1\right),\log_b\left(\left\lceil\frac{n}{b}\right\rceil\right)\right[$. En effet, si on avait un tel entier k, alors en composant par la fonction strictement croissante $x\mapsto b^x$, on aurait $\left\lceil\frac{n}{b}\right\rceil-1<b^k<\left\lceil\frac{n}{b}\right\rceil$ ce qui ne peut pas être.
Donc, nécessairement,
$$\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)-1\leqslant\log_b\left(\left\lceil\frac{n}{b}\right\rceil-1\right)<\log_b n-1\leqslant\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)$$
soit
$$\underbrace{\left(\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)+1\right)}_{\in\mathbb{Z}}-1<\log_b n\leqslant\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)+1$$
D'où, par unicité, $\boxed{\ell(n)=\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)+1.}$

1. D'après un sujet de A. Troesch

4. Il s'agit d'utiliser une inégalité de convexité classique due à la concavité du ln : $\forall\, x>-1,\ \ln(1+x)\leqslant x$ c'est-à-dire $\forall\, x>0,\ \ln x\leqslant x-1$ (la courbe du ln est sous sa tangente en 1).

   On a donc, pour $k\in\mathbb{N}$ $\boxed{\ln(P(k))=\sum_{i=1}^{k}\ln\left(1+\frac{b-1}{b^i}\right)\leqslant\sum_{i=1}^{k}\frac{b-1}{b^i}.}$

   Ainsi, $\ln(P(k))\leqslant\sum_{i=1}^{k}\left(\left(\frac{1}{b}\right)^{i-1}-\left(\frac{1}{b}\right)^{i}\right)=1-\left(\frac{1}{b}\right)^{k}\leqslant 1$ par télescopage. Alors $(\ln(P(k)))$ est majorée et donc $\boxed{(P(k))\text{ est majorée.}}$

---

5. Soit $n\geqslant b$. D'après la question 3,

$$Q_n=P(\ell(n)-1)=P\left(\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)\right)=\prod_{i=1}^{\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)}\left(1+\frac{b-1}{b^i}\right)=\left(1+\frac{b-1}{b^{\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)}}\right)P\left(\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)-1\right)=\left(1+\frac{b-1}{b^{\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)}}\right)Q_{\left\lceil\frac{n}{b}\right\rceil}.$$

   Or $\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)=\ell(n)-1\leqslant\log_b(n)$ donc $b^{\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)}\leqslant n$ et $\boxed{Q_n\geqslant\left(1+\frac{b-1}{n}\right)Q_{\left\lceil\frac{n}{b}\right\rceil}.}$

---

6. La définition $K=\max_{n\in[\![1,b-1]\!]}\left(\frac{T(n)}{n^\gamma Q_n^\gamma}-M(\ell(n)-1)\right)$ ne pose pas de problème car il y a un nombre fini de valeurs.

   Pour $n\in\mathbb{N}^*$, on pose $P(n)$: « $\frac{T(n)}{n^\gamma}\leqslant Q_n^\gamma(K+M(\ell(n)-1))$ ».

   On raisonne par récurrence forte. Par définition de K, on a déjà $P(n)$ vraie pour tout $n\in[\![1,b-1]\!]$.

   Soit $n\geqslant b$. On suppose $P(k)$ vraie pour tout $k\in[\![1,n-1]\!]$.

   On va utiliser la relation de récurrence définissant T en remarquant que $\left\lfloor\frac{n}{b}\right\rfloor<n$ et $\left\lceil\frac{n}{b}\right\rceil<n$:

$$T(n)=\alpha T\left(\left\lfloor\frac{n}{b}\right\rfloor\right)+\beta T\left(\left\lceil\frac{n}{b}\right\rceil\right)+u_n\leqslant\alpha\left\lceil\frac{n}{b}\right\rceil^\gamma Q^\gamma_{\left\lfloor\frac{n}{b}\right\rfloor}\left(K+M\left(\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)-1\right)\right)+\beta\left\lfloor\frac{n}{b}\right\rfloor^\gamma Q^\gamma_{\left\lceil\frac{n}{b}\right\rceil}\left(K+M\left(\ell\left(\left\lfloor\frac{n}{b}\right\rfloor\right)-1\right)\right)+Mn^\gamma$$

   avec $\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)=\ell(n)-1$ d'après 3 et $\ell\left(\left\lfloor\frac{n}{b}\right\rfloor\right)-1\leqslant\ell\left(\left\lceil\frac{n}{b}\right\rceil\right)-1=\ell(n)-2$ par croissance de ℓ.

   La question 2 nous dit aussi que $\left\lfloor\frac{n}{b}\right\rfloor\leqslant\left\lceil\frac{n}{b}\right\rceil\leqslant\frac{b+n-1}{b}$. Comme $a=\alpha+\beta$ et par croissance de (Q_n), il vient

$$T(n)\leqslant a\left(\frac{b+n-1}{b}\right)^\gamma Q^\gamma_{\left\lceil\frac{n}{b}\right\rceil}(K+M(\ell(n)-2))+Mn^\gamma.$$

   En utilisant la question précédente, et avec $b\geqslant 1$, $a=b^\gamma$, $Q_n^\gamma\geqslant 1$ par hypothèse, on tire

$$T(n)\leqslant b^\gamma\left(\frac{b+n-1}{b}\right)^\gamma\left(\frac{n}{b+n-1}\right)^\gamma Q_n^\gamma(K+M(\ell(n)-2))+Q_n^\gamma Mn^\gamma=n^\gamma Q_n^\gamma(K+M(\ell(n)-1)).$$

   ce qui établit la récurrence.

   Ainci, pour tout $n\in\mathbb{N}^*$. $\boxed{\frac{T(n)}{n^\gamma}\leqslant Q_n^\gamma(K+M(\ell(n)-1)).}$

---

7. On a alors $\frac{T(n)}{n^\gamma\log_b(n)}\leqslant Q_n^\gamma\frac{K+M(\ell(n)-1)}{\log_b(n)}$ qui est majoré car P l'est d'après 4 et $\frac{K+M(\ell(n)-1)}{\log_b(n)}\xrightarrow[n\to+\infty]{}M$ d'après 1. Donc $\boxed{T(n)=\mathcal{O}\left(n^\gamma\log_b(n)\right)}$ puis, comme $\log_b n=\frac{\ln n}{\ln b}$, $\boxed{T(n)=\mathcal{O}(n^\gamma\ln(n)).}$

E. Exemples

1. **L'algorithme d'exponentiation rapide**

   (a) Par récurrence forte, avec la propriété $P(n)$: « l'algorithme renvoie x^n ».

   $P(0)$ est vraie.

Soit $n \in \mathbb{N}^*$ tel que pour tout $k \in [\![0, n-1]\!]$, $P(k)$ est vraie. Alors $\left\lfloor \frac{n}{2} \right\rfloor < n$ donc l'algorithme renvoie bien $p = x^{\lfloor \frac{n}{2} \rfloor}$. Puis, si n est pair, il renvoie $p^2 = x^n$ et si n est impair, il renvoie $xp^2 = x\left(x^{\frac{n-1}{2}}\right)^2 = x^n$ ce qui nous permet de conclure que l'algorithme renvoie bien x^n.

(b) La description de l'algorithme donne pour $n \in \mathbb{N}$, un nombre borné d'opérations à effectuer pour calculer $\left\lfloor \frac{n}{2} \right\rfloor$, stocké le résultat de l'appel récursif dans p, puis calculer p^2 ou xp^2 suivant la parité de n.

On a donc bien (u_n) bornée telle que pour tout $n \in \mathbb{N}$, $T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + u_n$.

(c) On est ramené à l'étude précédente avec $b = 2$, $\alpha = 1$, $\beta = 0$, $\gamma = \log_2(1+0) = 0$ (on a bien $u_n = \mathcal{O}(1)$).

On obtient alors $T(n) = \mathcal{O}(\ln n)$.

(d) L'algorithme naïf ayant une complexité linéaire (en $\mathcal{O}(n)$), c'est bien meilleur ici.

2. **Le tri fusion**

La fusion s'effectue en temps linéaire : on va choisir à chaque étape l'élément suivant de la liste fusionnée en parcourant chaque élément des deux listes initiales.

On obtient une équation de complexité $T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + u_n$ avec $u_n = \mathcal{O}(n)$ ce qui, avec l'étude précédente, en posant $b = 2$, $\alpha = 1$, $\beta = 1$, $\gamma = \log_2(1+1) = 1$ (on a bien $u_n = \mathcal{O}(n)$).

On obtient alors $T(n) = \mathcal{O}(n \ln n)$.

Implémentation en Ocaml avec des listes :

OCaml

```
let rec division liste =
  (* 'a list -> 'a list * 'a list *)
  (* Renvoie deux sous-listes de taille moitié à +/- 1 près *)
  match liste with
  | [] -> [], []
  | tete :: [] -> liste, []
  | tete1 :: tete2 :: queue ->
  let liste1, liste2 = division queue in
    tete1 :: liste1, tete2 :: liste2
;;

let rec fusion liste1 liste2 =
  (* 'a list -> 'a list -> 'a list *)
  (* fusionne les deux listes supposées triées *)
  match liste1, liste2 with
  | [], _ -> liste2
  | _, [] -> liste1
  | tete1 :: queue1, tete2 :: queue2 ->
      if tete1 < tete2 then
          tete1 :: (fusion queue1 liste2)
      else
          tete2 :: (fusion liste1 queue2)
;;

let rec tri_fusion liste =
  (* 'a list -> 'a list *)
  match liste with
  | [] -> []
  | tete :: [] -> liste
  | _ -> let liste1, liste2 = division liste in
      fusion (tri_fusion liste1) (tri_fusion liste2)
;;
```

Implémentation en Ocaml avec des tableaux :

OCaml

```
let fusion_tab tab debut milieu fin =
  (* fusionne les parties de tab entre [debut, milieu] et [milieu + 1, fin] *)
  (* à l'aide d'un tableau temporaire *)
  let tableau = Array.make (fin - debut + 1) tab.(0) in
  let i1 = ref debut in
  let i2 = ref (milieu + 1)  in
  for k = 0 to (fin - debut )  do
     if !i1 < milieu + 1  && (!i2 = fin + 1  || tab.(!i1) < tab.(!i2)) then
         (* Cas où on prend un élément de la première partie du tableau *)
         begin
         tableau.(k) <- tab.(!i1);
         incr i1
         end
     else
         (* Cas où on prend un élément de la deuxième partie du tableau *)
         begin
         tableau.(k) <- tab.(!i2);
         incr i2
         end
  done;
  for k = debut to fin do
      tab.(k) <- tableau.(k - debut);
  done
;;

let tri_fusion_tab tableau =
  let rec tri_aux tab deb fin =
    if deb < fin then
    begin
    let mil = (deb + fin ) / 2  in
    tri_aux tab deb mil;
    tri_aux tab (mil + 1) fin;
    fusion_tab tab deb mil fin;
    end
  in
  tri_aux tableau 0 (Array.length tableau - 1)
;;
```

FIN DE L'ÉNONCÉ