Technique de résolution des équations de récurrence aux divisions finis

Andrey Martinez Cruz

Glossaire

Définition

Pour rappel, les équations de récurrence aux divisions finis sont des équations de la forme :

$$T(n) = \alpha T(\frac{n}{\beta}) + f(n) \tag{1}$$

où $\alpha \geq 1$, $\beta \geq 2$ et $f(n) : \mathbb{N} \rightarrow \mathbb{R}^+$.

Exemples

Les équations suivants sont des exemples d'équation de récurrence aux divisions finis :

- → $2T(\frac{n}{5}) + 1$

Exemples

Les équations suivants sont des exemples d'équation de récurrence aux divisions finis :

- → $2T(\frac{n}{5}) + 1$
- → $6T(\frac{n}{9}) + n$

Exemples

Les équations suivants sont des exemples d'équation de récurrence aux divisions finis :

- → $2T(\frac{n}{5}) + 1$
- → $6T(\frac{n}{9}) + n$
- → $4T(\frac{n}{2}) + \log n$

Méthodes de résolutions

Parmi les méthodes de résolutions possibles, il y a :

- Méthode itérative

Méthodes de résolutions

Parmi les méthodes de résolutions possibles, il y a :

- Méthode itérative
- Méthode en arbre

Méthodes de résolutions

Parmi les méthodes de résolutions possibles, il y a :

- Méthode itérative
- Méthode en arbre
- Théorème maître*

Quelques pré-requis

Voici des notion qui seront importantes pour faire l'analyse des équations de récurrence finis

Quelques pré-requis

Voici des notion qui seront importantes pour faire l'analyse des équations de récurrence finis :

- Calculer les sommations et reconnaître les sommations (surtout les sommes géométriques)
- Savoir manipuler des logarithmes (savoir utiliser cette régle : $a^{\log_b n} = n^{\log_b a}$ où a est une constante sera très utile).

Approche

Le but de la méthode itérative est de développer l'équation de récurrence afin de trouver une formule qui représente le motif.

Exemple

Supposons on a l'équation de récurrence suivante :

$$T(n) = 4T(\frac{n}{2}) + n \tag{2}$$

Le développement sera de cette équation de récurrence sera le suivante pour $n = 8$:

$$\begin{aligned} T(8) &= 4T(4) + f(8) \\ &= 4(4T(2) + f(4)) + f(8) \\ &= 4(4(4T(1) + f(2)) + f(4)) + f(8) \\ &= 4(4(4f(1) + f(2)) + f(8)) + f(8) \\ &= 4(16f(1) + 4f(2) + f(4)) + f(8) \\ &= 64f(1) + 16f(2) + 4f(4) + f(8) \end{aligned}$$

Substitution

En voyant un peu comment la fonction se comporte, on peut poser $n = 2^p$ et avoir la sommation suivante :

$$T(n) = \sum_{i=0}^{p} 2^{2^i} 2^{p-i} \tag{3}$$

Résolution

$$
\begin{aligned}
T(n) = \sum_{i=0}^{p} 2^{2i} 2^{p-i} &= 2^p \sum_{i=0}^{p} 2^i \\
&= 2^p \left(\frac{1 - 2^{p+1}}{1-2}\right) \\
&= 2^p(-1 + 2^{p+1}) \\
&= 2^{2p+1} - 2^p \\
&= 2n^2 - n \in \Theta(n^2)
\end{aligned}
$$

Résolution

Supposant que l'on veut évaluer la complexité de cette équation de récurrence :

$$T(n) = 3T(\frac{n}{2}) + n \tag{4}$$

Résolution

Supposant que l'on veut évaluer la complexité de cette équation de récurrence :

$$T(n) = 3T(\frac{n}{2}) + n \tag{4}$$

Comment peut-on résoudre cette équation de récurrence ?

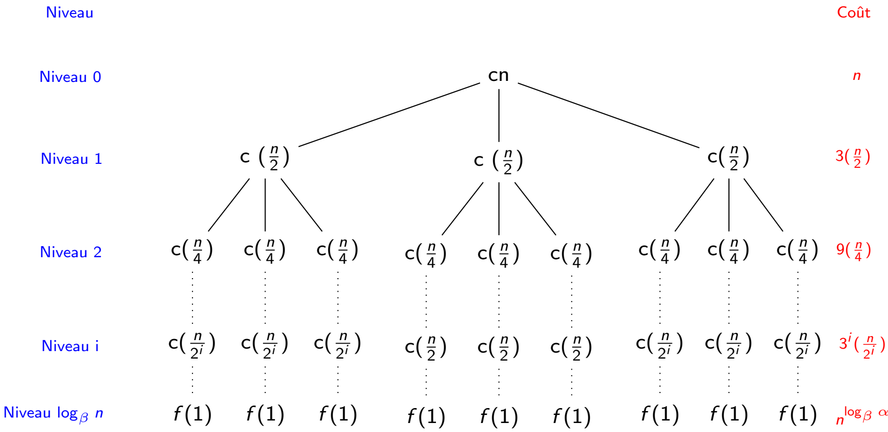
Résolution

Supposant que l'on veut évaluer la complexité de cette équation de récurrence :

$$T(n) = 3T(\frac{n}{2}) + n \tag{4}$$

Comment peut-on résoudre cette équation de récurrence ? Au lieu de deviner, on peut faire un dessin des appels pour voir ce qu'il se passe.

Arbre


Niveau
Coût
Niveau 0
cn
n
Niveau 1
$c\left(\frac{n}{2}\right)$ $c\left(\frac{n}{2}\right)$ $c\left(\frac{n}{2}\right)$
$3\left(\frac{n}{2}\right)$
Niveau 2
$c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$ $c\left(\frac{n}{4}\right)$
$9\left(\frac{n}{4}\right)$
Niveau i
$c\left(\frac{n}{2^i}\right)$ $c\left(\frac{n}{2^i}\right)$ $c\left(\frac{n}{2^i}\right)$ $c\left(\frac{n}{2}\right)$ $c\left(\frac{n}{2}\right)$ $c\left(\frac{n}{2}\right)$ $c\left(\frac{n}{2^i}\right)$ $c\left(\frac{n}{2^i}\right)$ $c\left(\frac{n}{2^i}\right)$
$3^i\left(\frac{n}{2^i}\right)$
Niveau $\log_\beta n$
$f(1)$ $f(1)$ $f(1)$ $f(1)$ $f(1)$ $f(1)$ $f(1)$ $f(1)$ $f(1)$
$n^{\log_\beta \alpha}$

Travail total

Pour obtenir, le travaille totale de l'arbre, il faut fait la sommation de du niveau maximale possible à atteindre plus le coût total de tous les noeuds qui ont été explorer.

Travail total

Pour obtenir, le travaille totale de l'arbre, il faut fait la sommation de du niveau maximale possible à atteindre plus le coût total de tous les noeuds qui ont été explorer.
En bref, dans ce cas-ci,

$$T(n) = n^{\log_2 3} + \sum_{i=0}^{\log_2 n-1} (\frac{3}{2})^i n \tag{5}$$

Maintenant, on peut faire une analyse classique de ce dernier.

Résolution

Posons que $n = 2^p$

$$T(n) = n^{\log_2 3} + \sum_{i=0}^{\log_2 n-1} (\frac{3}{2})^i n = 2^{p \log_2 3} + \sum_{i=0}^{p} (\frac{3}{2})^i 2^p$$

$$= 2^{p \log_2 3} + 2^p \sum_{i=0}^{p-1} (\frac{3}{2})^i$$

$$= 2^{p \log_2 3} + 2^p (\frac{1-(3/2)^p}{1-(3/2)})$$

$$= n^{\log_2 3} + n(\frac{1-n^{\log_2 \frac{3}{2}}}{-0,5})$$

$$= n^{\log_2 3} + n(-2 + \frac{n^{\log_2 \frac{3}{2}}}{0,5}) = n^{\log_2 3} - 2n + \frac{n^{\log_2 \frac{3}{2}+1}}{0,5}$$

$$= n^{\log_2 3} - 2n + 2n^{\log_2 3} = 3n^{\log_2 3} - 2n$$

$$T(n) \in \Theta(n^{\log_2 3})$$

À date

Ce qu'on a fait c'est de développer l'équation et

À date

Ce qu'on a fait c'est de développer l'équation et"deviner" sa complexité

À date

Ce qu'on a fait c'est de développer l'équation et"deviner" sa complexité

Existe t'il un moyen de résoudre de façon exacte sans jouer à la roulette russe et sans dessiner un arbre ?

Théorème maître

Pour une équation de récurrence de la forme suivante :

$$T(n) = \alpha T(\frac{n}{\beta}) + f(n) \tag{6}$$

On a le théorème suivant :

Théorème

Soient $c = \log_\beta \alpha$ et on aboutit à un des ces trois cas :

1. Si $f(n) \in \mathcal{O}(n^{c-\epsilon})$pour un $\epsilon > 0$, alors $T(n) \in \Theta(n^c)$.
2. Si $f(n) \in \Theta(n^c \log^k(n))$ pour un $k \geq 0$, alors $T(n) \in \Theta(n^c \log^{k+1} n)$.
3. Si $f(n) \in \Omega(n^{c+\epsilon})$ pour un $\epsilon > 0$ et s'il existe un c telle que $0 < c < 1$ et $n_0 \in \mathbb{N}$ tels que $\forall n, n \geq n_0$, $\alpha f(\frac{n}{\beta}) \leq cf(n)$, alors $T(n) \in \Theta(f(n))$.

Raccourci

Pour $c = \log_\beta \alpha$ et $f(n) \in \Theta(n^\lambda)$, alors en comparant c et λ, on peut aboutir à un de ces cas trois cas :

Cas 1

Si $c > \lambda$, alors $T(n) \in \Theta(n^c)$.

Cas 2

Si $c = \lambda$, alors $T(n) \in \Theta(n^c \log n)$

Cas 3

Si $c < \lambda$, alors $T(n) \in \Theta(n^\lambda)$

Traduction

Les cas de théorèmes maître peuvent être formulés ainsi :

Traduction

Les cas de théorèmes maître peuvent être formulés ainsi :

- Cas 1 : le coût des appels récursifs est plus importante que la reconstruction de la solution des sous-problèmes.

Traduction

Les cas de théorèmes maître peuvent être formulés ainsi :

- Cas 1 : le coût des appels récursifs est plus importante que la reconstruction de la solution des sous-problèmes.
- Cas 2 : le coût des appels récursifs est équivalent au coût de reconstruction de la solution des sous-problèmes.

Traduction

Les cas de théorèmes maître peuvent être formulés ainsi :

- Cas 1 : le coût des appels récursifs est plus importante que la reconstruction de la solution des sous-problèmes.
- Cas 2 : le coût des appels récursifs est équivalent au coût de reconstruction de la solution des sous-problèmes.
- Cas 3 : le coût de reconstruction est plus importante que les appels récursifs.

Example 1

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 2T(n/2) + \sqrt{n} \tag{7}$$

Example 1

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 2T(n/2) + \sqrt{n} \tag{7}$$

$\alpha = 2$, $\beta = 2$ et $\lambda = \frac{1}{2}$. $c = \log_\beta \alpha = \log_2 2 = 1$ et $c > \lambda$, donc on est dans le premier cas du théorème maître et donc $T(n) \in \Theta(n)$.

Example 2

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 3T(n/4) + n^2 \tag{8}$$

Example 2

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 3T(n/4) + n^2 \tag{8}$$

$\alpha = 3$, $\beta = 4$ et $\lambda = 2$. $c = \log_\beta \alpha = \log_4 3$ et $c < \lambda$, donc on pourrait être dans le troisième cas du théorème maître.

Example 2

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 3T(n/4) + n^2 \tag{8}$$

$\alpha = 3$, $\beta = 4$ et $\lambda = 2$. $c = \log_\beta \alpha = \log_4 3$ et $c < \lambda$, donc on pourrait être dans le troisième cas du théorème maître.

$$3(\frac{n}{4})^2 \leq cn^2$$

$$3(\frac{n^2}{16}) \leq cn^2$$

$$\frac{3}{16}n^2 \leq cn^2$$

Example 2

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 3T(n/4) + n^2 \tag{8}$$

$\alpha = 3$, $\beta = 4$ et $\lambda = 2$. $c = \log_\beta \alpha = \log_4 3$ et $c < \lambda$, donc on pourrait être dans le troisième cas du théorème maître.

$$3(\frac{n}{4})^2 \leq cn^2$$

$$3(\frac{n^2}{16}) \leq cn^2$$

$$\frac{3}{16}n^2 \leq cn^2$$

Cela fonctionne pour $c = 0,9$ et $\epsilon = 2$ telle que $n^2 \in \Omega(n^{c+\epsilon})$. Donc, $T(n) \in \Theta(n^2)$.

Example 3

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 8T(n/2) + n^3 \tag{9}$$

Example 3

Donner la complexité de l'équation de récurrence suivante en utilsant le théorème maître :

$$T(n) = 8T(n/2) + n^3 \tag{9}$$

$\alpha = 8$, $\beta = 2$ et $\lambda = 3$. $c = \log_\beta \alpha = \log_2 8 = 3$ et $c = \lambda$, donc on est dans le deuxième cas du théorème maître et donc $T(n) \in \Theta(n^3 \log n)$.

Problème de raccourci

Quelle est la complexité de cette équation de récurrence :

$$T(n) = 2T(\frac{n}{2}) + n \log n \tag{10}$$

À première vue, on pourrait penser que le cas 3 s'appliquer, car $\alpha = 2$, $\beta = 2$ et donc, $c = \log_2 2 = 1$ et donc $n^c = n \in o(n \log n)$. Voyons si cela est vrai :

$$\begin{aligned}
2(\frac{n}{2} \log \frac{n}{2}) &\leq c(n \log n) \\
n(\log \frac{n}{2}) &\leq c(n \log n) \\
n(\log n - \log 2) &\leq c(n \log n) \\
n \log n - n &\leq c(n \log n) \\
\log n - 1 &\leq c \log n \\
1 - \frac{1}{\log n} &\leq c
\end{aligned}$$

Ici, le cas 3 du théorème échoue, car il faudraut que $c \geq 1$, mais $0 < c < 1$ et donc, $T(n) \notin \Theta(f(n))$. La bonne complexité : $T(n) \in \Theta(n \log^2 n)$.