. RELAŢII RECURENTE

Cel mai important câştig al exprimării recursive este faptul că ea este naturală şi compactă, fără să ascundă esenţa algoritmului prin detaliile de implementare. Pe de altă parte, apelurile recursive trebuie folosite cu discernământ, deoarece solicită şi ele resursele calculatorului (timp şi memorie). Analiza unui algoritm recursiv implică rezolvarea unui sistem de recurente. Vom vedea în continuare cum pot fi rezolvate astfel de recurente.

Când un algoritm conţine o apelare recursivă la el însuşi, timpul său de execuţie poate fi descris adesea printr-o recurenţă. O *recurenţă* este o ecuaţie sau o inegalitate care descrie întregul timp de execuţie al unei probleme de dimensiune *n* cu ajutorul timpilor de execuţie pentru date de intrare de dimensiuni mici. Putem, apoi, folosi instrumente matematice pentru a rezolva problema de recurenţă şi pentru a obţine margini ale performanţei algoritmului.

O recurenţă pentru timpul de execuţie al unui algoritm de tipul divide şi stăpâneşte se bazează pe cele trei etape definite în descrierea metodei. La fel ca până acum, vom nota cu $T(n)$ timpul de execuţie al unei probleme de dimensiune n. Dacă dimensiunea problemei este suficient de mică, de exemplu $n \le c$ pentru o anumită constantă c, soluţia directă ia un timp constant de execuţie, pe care îl vom nota cu $\Theta(1)$. Să presupunem că divizăm problema în a subprobleme, fiecare dintre acestea având dimensiunea de $1/b$ din dimensiunea problemei originale. Dacă *D(n)* este timpul necesar pentru a divide problema în suprobleme, iar *C(n)* este timpul necesar pentru a combina soluţiile subproblemelor în soluţia problemei originale, obţinem recurenţa

$$T(n) = \begin{cases} \Theta(n), & n \le c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n), & n > c \end{cases}$$

5.1 Metoda master

Metoda master furnizează o „reţetă" pentru rezolvarea recurenţelor de forma

$$T(n) = aT(n/b) + f(n) \qquad (5.1)$$

unde $a \ge 1$ şi $b > 1$sunt constante, iar $f(n)$ este o funcţie asimptotic pozitivă. Metoda master pretinde memorarea a trei cazuri, dar apoi soluţia multor recurenţe se poate determina destul de uşor, de multe ori fără creion şi hârtie.

Recurenţa (1.4.1) descrie timpul de execuţie al unui algoritm care împarte o problemă de dimensiune n în a subprobleme, fiecare de dimensiune n/b, unde a şi b sunt constante pozitive. Cele a subprobleme sunt rezolvate recursiv, fiecare în timp $T(n/b)$. Costul divizării problemei şi al combinării rezultatelor subproblemelor este descris de funcţia *f(n)* (Adică, utilizând notaţia $f(n) = D(n) + C(n)$). Din punctul de vedere al corectitudinii tehnice, recurenţa nu este, de fapt, bine definită, deoarece n/b ar putea să nu fie întreg. Înlocuirea fiecăruia dintre cei a termeni $T(n/b)$ fie cu $T(\lfloor n/b \rfloor)$ fie cu $T(\lceil n/b \rceil)$ nu afectează, totuşi, comportamentul asimptotic al recurenţei. În mod normal, ne va conveni, de aceea, să omitem funcţiile parte întreagă inferioară şi superioară când scriem recurenţe divide şi stăpâneşte de această formă.

Teorema master

Metoda master depinde de următoarea teoremă.

Teorema 5.1.1. **(***teorema master***)** Fie $a \ge 1$ şi $b > 1$ constante, fie $f(n)$ o funcţie şi fie $T(n)$ definită pe întregii nenegativi prin recurenţă

$$T(n) = aT(n/b) + f(n)$$

unde interpretăm n/b fie ca $\lfloor n/b \rfloor$ fie ca $\lceil n/b \rceil$. Atunci $T(n)$ poate fi delimitată asimptotic după cum urmează.

1. Dacă $f(n) = O\left(n^{\log_b a - \varepsilon}\right)$ pentru o anumită constantă $\varepsilon > 0$, atunci $T(n) = \Theta\left(n^{\log_b a}\right)$.
2. Dacă $f(n) = \Theta\left(n^{\log_b a}\right)$, atunci $T(n) = \Theta\left(n^{\log_b a} \lg n\right)$.
3. Dacă $f(n) = \Omega\left(n^{\log_b a + \varepsilon}\right)$, pentru o anumită constantă $\varepsilon > 0$ şi dacă $af(n/b) \le cf(n)$ pentru o anumită constantă $c < 1$şi toţi n suficient de mari, atunci $T(n) = \Theta(f(n))$.

Înainte de a aplica teorema master la câteva exemple, să ne oprim puţin ca să înţelegem ce spune. În fiecare dintre cele trei cazuri, comparăm funcţia $f(n)$ cu funcţia $n^{\log_b a}$. Intuitiv, soluţia recurenţei este determinată de cea mai mare dintre cele două funcţii. Dacă funcţia $n^{\log_b a}$ este mai mare, ca în cazul 1, atunci soluţia este $T(n) = \Theta\left(n^{\log_b a}\right)$. Dacă

funcţia $f(n)$ este mai mare, ca în cazul 3, atunci soluţia este $T(n)=\Theta(f(n))$. Dacă cele două funcţii sunt de acelaşi ordin de mărime, ca în cazul 2, înmulţim cu un factor logaritmic, iar soluţia este

$$T(n)=\Theta\left(n^{\log_b^a}\lg n\right)=\Theta(f(n)\lg n).$$

Dincolo de această intuiţie, există nişte detalii tehnice care trebuie înţelese.

În primul caz, $f(n)$ trebuie nu numai să fie mai mică decât $n^{log_b a}$, trebuie să fie polinomial mai mică. Adică $f(n)$ trebuie să fie asimptotic mai mică decât $n^{log_b a}$ cu un factor n^{ε} pentru o anumită constantă $\varepsilon>0$. În al treilea caz, $f(n)$ trebuie nu numai să fie mai mare decât $n^{log_b a}$, trebuie să fie polinomial mai mare şi, în plus, să verifice condiţia de „regularitate” $af(n/b)\le cf(n)$. Această condiţie este satisfăcută de majoritatea funcţiilor polinomial mărginite pe care le vom întâlni.

Este important de realizat că cele trei cazuri nu acoperă toate posibilităţile pentru $f(n)$.

Există un gol între cazurile 1 şi 2 când $f(n)$ este mai mic decât $n^{log_b a}$ dar nu polinomial mai mic. Analog există un gol între cazurile 2 şi 3 când $f(n)$ este mai mare decât $n^{log_b a}$ dar nu polinomial mai mare. Dacă funcţia $f(n)$ cade într-unul dintre aceste goluri, sau când condiţia de regularitate din cazul 3 nu este verificată, metoda master nu poate fi utilizată pentru a rezolva recurenţa.

Exemplu 5.1.1.

$T(n) = 9T(n/3) + n.$

Pentru această recurenţă, avem $a = 9$, $b = 3$, $f(n) = n$ şi astfel $n^{\log_b a}= T(n)= n^{\log_3 9}=\Theta(n^2)$.

Deoarece $f(n)= O(n^{\log_3 9-\varepsilon})$, unde $\varepsilon = 1$, putem să aplicăm cazul 1 al teoremei master şi să considerăm că soluţia este $T(n)= \Theta(n^2)$.

Exemplu 5.1.2.

$T(n)=T(2n/3) + 1.$

Pentru această recurenţă, avem $a=1$, $b= 3/2$, $f(n)=1$ şi $n^{\log_b a}= n^{\log_{3/2} 1}= n^0 =1$. Cazul 2 este cel care se aplică deoarece $f(n)= \Theta(n^{\log_b a})= \Theta(1)$ şi astfel soluţia recurenţei este

$T(n)= \Theta(\lg n).$

Exemplu 5.1.3.

$T(n) = 3T(n/4) + n\lg n,$

Pentru această recurenţă, avem $a=3$, $b=4$, $f(n)= n\lg n$ şi $n^{\log_b a}=n^{\log_4 3} =O(n^{0,793})$. Deoarece $f(n)=\Omega(n^{\log_4 3+\varepsilon})$, unde $\varepsilon \approx 0.2$, cazul 3 se aplică dacă putem arăta că pentru $f(n)$ este verificată condiţia de regularitate. Pentru n suficient de mare,

$af(n/b)=3(n/4)\lg (n/4) \le (3/4)n\lg n = cf(n)$ pentru $c = 3/4$.

În consecinţă, din cazul 3, soluţia recurenţei este $T(n)= \Theta(n\lg n)$.

Exemplu 5.1.4.

Metoda master nu se aplică recurenţei

$$T(n)= 2T(n/2) + n\lg n,$$

chiar dacă are forma potrivită: $a=2$, $b=2$, $f(n)= n\lg n$ şi $n^{\log_b a} = n$. Se pare că ar trebui să se aplice cazul 3, deoarece $f(n)= n\lg n$ este asimptotic mai mare decât $n^{\log_b a} = n$, dar nu polinomial mai mare. Raportul $f(n)/n^{\log_b a} = (n\lg n)/n$ este asimptotic mai mic decât n^{ε} pentru orice constantă pozitivă ε. În consecinţă, recurenţa cade în golul dintre cazurile 2 şi 3.