Cap.1. Descrierea şi evaluarea algoritmilor

Prof.dr.ing. Gabriela Ciuprina

Universitatea Politehnica Bucureşti, Facultatea de Inginerie Electrică,
Departamentul de Electrotehnică

Suport didactic pentru disciplina *Metode numerice în ingineria electrică, 2017-2018*

Cuprins

Algoritm

Metodă de rezolvare a unei probleme bazată pe descompunerea în etape simple, elementare.

Pseudocod

Metodă de descriere a algoritmilor.

- fără sintaxă strictă;
- clar, neambiguu;
- cuvinte cheie (în limba română.

Alcătuit din doua feluri de linii

- declaraţii - descriu datele;
- instrucţiuni - descriu acţiunile.

Tipuri de date

Fundamentale (simple)

logic, întreg, real, caracter

```
logic T, F          ; adevărat (true - în engl.), fals
întreg N            ; numărul de noduri
real Pc, Pg         ; putere consumată, putere generată
caracter c, C
```

Agregate

tablou, înregistrare

```
tablou real V[10]
                        V(1), V(5)    sau
întreg N                V1, V5
tablou real V[N]
```

Tipuri de date

Fundamentale (simple)

logic, întreg, real, caracter

```
logic T, F          ; adevărat (true - în engl.), fals
întreg N            ; numărul de noduri
real Pc, Pg         ; putere consumată, putere generată
caracter c, C
```

Agregate

tablou, înregistrare

```
înregistrare punct
        logic polar
        real coord1
        real coord2
```

punct.polar
punct.coord1
punct.coord2

Instrucţiuni de intrare/ieşire

`citeşte` *N*
`citeşte` *q*, *p* ; se admite citirea unei liste de variabile
`scrie` *N*
`scrie` *q*, *p* ; se admite scrierea unei liste de variabile

Instrucţiunea de atribuire

`logic` a	; rezultatul evaluării expresiei logice
`logic` b, c	; operanzi logici
`real` x, y	; operanzi aritmetici
$a =$ (b `sau` (`nu` c))	; expresie logică cu operatori logici
$a = (x \leq y)$	; expresie logică cu operatori de relaţie
$a = (x = y)$	; expresie logică cu operatori de relaţie

`întreg` i

`real` d, x, y

$i = i + 1$

$d = xy + \sin(y)$

$d = \sqrt{d}$

$$d = \frac{\frac{d^2}{2} + \frac{d}{3}}{\frac{1}{d}}$$

Decizia fără alternativă

```
real x
x = 2
dacă x < 0
        x = x + 3
        x = x/2
        x = x^2
x = 2x
```

```
real x
x = 2
dacă x < 0
        x = x + 3
        x = x/2
x = x^2
x = 2x
```

```
real x
x = 2
dacă x < 0
        x = x + 3
        x = x/2
        x = x^2
•
x = 2x
```

Decizia cu alternativă

```
real x                    ; un număr real - dată de intrare
real modul                ; dată de ieşire - modulul numărului real
citeşte x
dacă x ≥ 0
        modul = x
altfel
        modul = −x
•
scrie modul
```

Ciclul cu test initial

întreg N
întreg i
tablou real $x[N]$
$N = 3$
$x_1 = 1$
$x_2 = -1$
$x_3 = 2$
$i = 1$
cât timp $(i \geq 1)$ şi $(i \leq N)$
    dacă $x_i \geq 0$
        scrie "Elementul ", i ," este pozitiv"
    •
    i = i + 1
•

Ciclul cu test final

întreg N
tablou real $x[N]$
real s, ε
întreg k
...
$k = 0$
$s = 0$
repetă
    $k = k + 1$
    $s = s + x_k$
cât timp $|x_k| \geq \varepsilon$

Ciclul cu test final

întreg N
tablou real $x[N]$
real s, ε
întreg k
...
$k = 0$
$s = 0$
repetă
    $k = k + 1$
    $s = s + x_k$
până când $|x_k| < \varepsilon$

Ciclul cu contor

```
pentru contor = val_in, val_fin[, pas] [repetă]
        secvenţă
```

```
întreg N
citeşte N
tablou real a[N, N]
întreg i, j
pentru i = 1, N
        pentru j = 1, N
                citeşte a_{i,j}
```

Rutine: proceduri

Definiţie:

`procedură` nume_proc (lista argumentelor formale de I/O)
; comentarii ce descriu ce face procedura şi parametrii acesteia
. . .
; declaraţii pentru argumente
. . .
; instrucţiuni
. . .
`retur` ; se comandă întoarcerea în punctul de apel

Apel:

nume_proc (lista argumentelor actuale de intrare şi ieşire)

Rutine: funcţii

Definiţie:

`funcţie` nume_fct (lista argumentelor formale de intrare)
; comentarii ce descriu ce face funcţia şi parametrii acesteia
...
; declaraţii pentru argumente
...
; instrucţiuni
...
`întoarce` valoare ; se comandă întoarcerea în punctul de apel

Apel:

val = nume_fct (lista argumentelor actuale de intrare)

Exemplu

```
; program principal
întreg N
citeşte N
tablou real a[N], b[N]
real p
citeşte_vector(N,a)
citeşte_vector(N,b)
p = produs_scalar(N,a,b)
scrie p

procedură citeşte_vector(N,x)
întreg N
tablou real x[N]
întreg i
pentru i = 1, N
        citeşte x_i
•

funcţie produs_scalar(N,v,w)
întreg N
tablou real v[N], w[N]
întreg i
real r
r = 0
pentru i = 1, N
        r = r + v_i w_i
•
întoarce r
```

T = O(?)

Complexitatea unui algoritm din punct de vedere al timpului de calcul

relaţia dintre timpul de calcul exprimat în număr de operaţii elementare şi dimensiunea problemei

Operaţie elementară

operaţia care durează cel mai mult.

Algoritmi polinomiali - de ordinul 1 (liniar)

p = 0;
pentru $i = 1, n$
$p = p + a_i b_i$ $T = O(2n) \approx O(n)$

•

Algoritmi polinomiali - de ordinul 2 (pătratic)

- $\underline{\texttt{pentru}}\ i = 1, n$
  - $b_i = 0$
  - $\underline{\texttt{pentru}}\ j = 1, n$
    - $b_i = b_i + a_{ij} x_j$
  - •
- •

$T = O(2n^2) \approx O(n^2)$

Algoritmi polinomiali - de ordinul 3 (cubic)

$\underline{\texttt{pentru}}\ i = 1, n$
$\quad \underline{\texttt{pentru}}\ j = 1, n$
$\quad\quad c_{ij} = 0$
$\quad\quad \underline{\texttt{pentru}}\ k = 1, n$
$\quad\quad\quad c_{ij} = c_{ij} + a_{ik} b_{kj}$
$\quad\quad \bullet$
$\quad \bullet$
$\bullet$

$$T = O(2n^3) \approx O(n^3)$$

Algoritmi polinomiali - de ordinul k

$$T = O(n^k) \iff (\exists)\ C > 0 \text{ şi } n_0 \text{ a.i. } T \leq Cn^k\ (\forall)\ n \geq n_0$$

Algoritm de ordin 0: $T = O(1)$ - nu depinde de dimensiunea problemei.

$$O(1) < O(\log n) < O(n) < O(n \log n) < O(n^2) < \ldots O(e^n) < O(n!)$$

Ex1: Evaluarea unui polinom - varianta I

$$P(x) = a_0 + a_1 x + \ldots + a_n x^n. \quad (1)$$

$$P(x) = a_0 + \sum_{i=1}^{n} a_i x^i \quad (2)$$

; Varianta 1
$P = a_0$;
pentru $i = 1, n$
    $t = a_i$
    pentru $j = 1, i$
        $t = t * x$
    •
    P = P + t;
•

Nr. de operaţii elementare:
$\sum_{i=1}^{n}(i+1) \approx n^2/2$

$T_1 = O(n^2/2) \approx O(n^2)$

Ex1: Evaluarea unui polinom - varianta II

$$P(x) = a_0 + a_1 t_1 + a_2 t_2 + \ldots + a_n t_n = a_0 + \sum_{i=1}^{n} a_i t_i, \qquad (3)$$

unde

$$t_1 = x, \quad t_i = t_{i-1} x, \quad i = 2, \ldots, n. \qquad (4)$$

; Varianta 2
$P = a_0$;
$t = 1$;
pentru $i = 1, n$ $\qquad T_2 = \mathrm{O}(3n) \approx \mathrm{O}(n)$
    $t = t * x$
    $P = P + a_i * t$
•

Ex1: Evaluarea unui polinom - varianta III

$$P(x) = a_0 + x(a_1 + x(a_2 + \ldots x(a_{n-1} + a_n x)\ldots)). \qquad (5)$$

; Varianta 3
$P = a_n$;
pentru $i = n-1, 0, -1$ $\qquad T_3 = O(2n) \approx O(n)$
$\qquad P = a_i + P * x$

•

Ex2: Reducerea timpului de calcul

; Varianta A
întreg i, j, n
real a, b
...
tablou real $c[n][n]$
pentru $i = 1, n$
    pentru $j = 1, n$
        $c_{ij} = f(i * a) + f(j * b)$; f definită în altă parte
    •
•

Operaţie elementară: evaluarea funcţiei $f \Rightarrow T_A = \mathrm{O}(2n^2)$.

Ex2: Reducerea timpului de calcul

; Varianta B
întreg i, j, n
real a, b, p
...
tablou real $c[n][n]$
pentru $i = 1, n$
    $p = f(i * a)$
    pentru $j = 1, n$
        $c_{ij} = p + f(j * b)$
    •
•

$$T_B = O(n(n+1)) = O(n^2 + n) \approx O(n^2).$$

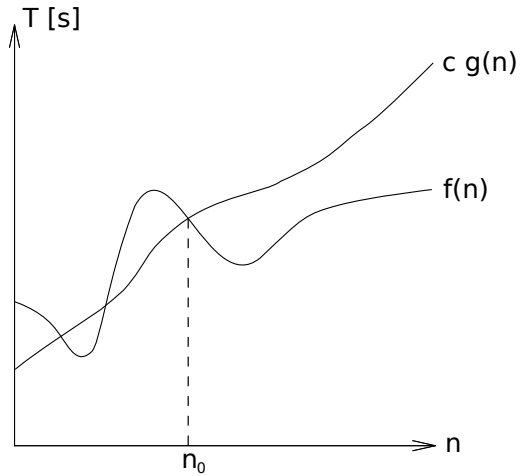
Ex2: Reducerea timpului de calcul

; Varianta C
întreg i, j, n
real a, b
...
tablou real $c[n][n]$
tablou real $p[n], q[n]$
pentru $i = 1, n$
    $p_i = f(i * a)$
    $q_i = f(i * b)$

$$T_C = O(2n)$$

•
pentru $i = 1, n$
    pentru $j = 1, n$
        $c_{ij} = p_i + q_j$
    •
•

Notaţie asimptotică O



$$T = f(n) = \mathrm{O}(g(n))$$

$\mathrm{T} = \mathrm{O}(g(n)) \iff \exists\ C > 0$ şi n_0 a.i. $T \leq Cg(n)$ $(\forall)$ $n \geq n_0$

M = O(?)

Complexitatea unui algoritm din punct de vedere al necesarului de memorie

relaţia dintre necesarul de memorie exprimat în număr de locaţii elementare şi dimensiunea problemei

Locaţia elementară de memorie

locaţia corespunzătoare unui număr real.

$M = O(n^k) \iff (\exists)\ C$ a.i. $M \leq Cn^k$.

Ex2: Reducerea timpului de calcul

; Varianta A
întreg i, j, n
real a, b
...
tablou real $c[n][n]$
pentru $i = 1, n$
    pentru $j = 1, n$
        $c_{ij} = f(i * a) + f(j * b)$; f definită în altă parte
    •
•

$T_A = O(2n^2)$
$M_A = O(n^2 + 2) \approx O(n^2)$

Ex2: Reducerea timpului de calcul

; Varianta B
întreg i, j, n
real a, b, p
...
tablou real $c[n][n]$
pentru $i = 1, n$
    $p = f(i * a)$
    pentru $j = 1, n$
        $c_{ij} = p + f(j * b)$
    •
•

$T_B = O(n(n+1)) = O(n^2 + n) \approx O(n^2)$
$M_B = O(n^2 + 3) \approx O(n^2)$

Ex2: Reducerea timpului de calcul

; Varianta C
întreg i, j, n
real a, b
...
tablou real $c[n][n]$
tablou real $p[n], q[n]$
pentru $i = 1, n$
$p_i = f(i * a)$
$q_i = f(i * b)$

$T_C = O(2n)$
$M_C = O(n^2 + 2n + 2) \approx O(n^2)$

•
pentru $i = 1, n$
pentru $j = 1, n$
$c_{ij} = p_i + q_j$
•
•

Concluzii

- Elaborarea unui algoritm nu se face în faţa calculatorului. De aceea este nevoie de un pseudolimbaj (pseudocod).
- Pseudocodul prezentat este cel pe care îl veţi întâni în îndrumar. Puteţi să îl alteraţi uşor, fără a fi penalizaţi, dacă el rămâne clar şi neambiguu.
- Elaborarea unui algoritm înseamnă întotdeauna stabilirea unui compromis între timp de calcul şi necesar de memorie.

Temă - pentru bonus

a) Propuneţi o structură de date care să descrie un circuit rezistiv liniar de curent continuu. (Indicaţie: amintiţi-vă cum erau descrise astfel de circuite în limbaj Spice).
b) Scrieţi pseudocodul unei proceduri care să aibă ca argument de intrare numele unui fişier de tip Spice şi ca argument de ieşire structura de date pe care aţi conceput-o anterior.
c) Evaluaţi complexitatea pseudocodului din punct de vedere al necesarului de memorie şi din punct de vedere al timpului de calcul, alegându-vă în mod potrivit: unul sau mai mulţi parametri de care depinde complexitatea procedurii şi operaţia de referinţă.

Tema nu se va redacta electronic. Tema se va preda la curs. Vor primi bonus pe ea primii 5 studenti care vor preda o rezolvare completă, corectă în proporţie de mai mult de 80 %.

Lectura obligatorie pentru această săptămână

- Pseudocod şi complexitate - Cap.1 din

  [1] Gabriela Ciuprina, Mihai Rebican, Daniel Ioan - Metode numerice in ingineria electrica - Indrumar de laborator pentru studentii facultatii de Inginerie electrica, Editura Printech, 2013, disponibil la

  http://mn.lmn.pub.ro/indrumar/IndrumarMN_Printech2013.pdf

- Recapitularea noţiunilor de Matlab - Cap.1 din
  [2] Gabriela Ciuprina - Algoritmi numerici prin exercitii si implementari in Matlab, Editura MatrixROM, 2013, disponibil la

  http://www.lmn.pub.ro/∼gabriela/books/AlgNrExMatlab_MatrixRom2013.pdf

Important: nu aveţi voie la acest curs/lab să folosiţi operaţii cu vectori şi matrice decât pentru validarea rezultatelor, nu pentru implementarea procedurilor. Pentru analiza complexitaţii este important să fie vizibile şi să fiţi conştienţi de toate operaţiile elementare ale algoritmilor implementaţi.