Numeričke metode

Aleksandar Maksimović
IRB

Sadržaj

- Programiranje u C i Fortranu, rezultati
- Karakteristike i razlike Fortrana i C
- Miješano programiranje
- Makefile
- Polja u memoriji

Uvod

- Podijeljeni smo u izboru jezika
  - 4 glasa C (u kombinaciji C++ i Mathematica)
  - 4 glasa F77
  - 1 glas neiskorišten
- Fortran je dugo vremena bio standard za matematičko programiranje
- Puno software-a napisan u F77 (npr. BLAS, LAPACK)
- C/C++ postaje dominantan jezik
- C i F77 mogu se povezati preko objektnog koda

Razlike f77 i C jezika

C jezik razlikuje mala i velika slova, npr. SUM i sum nije isto. Kod f77 u pitanju je ista funkcija. C ima samo funkcije. void funkcija odgovara podprogramima u f77. Kompajler f77 dodaje znak _ na ime funkcije.

C	*F77*
Nastavak je .c	Nastavak je .f ili .f77
Linija se terminira znakom ;	Nema posebnog znaka za kraj linije
Funkcije su definirane zagradama { ... }	Funkcije (podprogrami) završavaju END komandom
Slobodan format	Kod počinje u 7 koloni
Argumenti funkcija "by value" (kopiranje)	Argumenti funkcija "by reference"
Dinamičko alociranje memorije	Samo statičko alociranje
Indeksi polja počinju od 0	Indeksi polja počinju od 1
Matrice se spremaju "row major"	Matrice se spremaju "column major"
Komentar se nalazi između /* i */	Komentar počinje slovom c u prvoj koloni

tipovi varijabli

C	F77
int	integer
float	real
double	double precision
char	character

ekvivalentne naredbe kod F77 i C

Fortran 77	C

c ----------
c AUTHOR: L.M. Sophistis
c ----------
program main
. . . . . .
end
	/* ----------
	AUTHOR: L.M. Sophistis
	---------- */
	int main()
	{
	. . . . . .
	return 0;
	}

Dimension a(0:29),b(0:4)
Double precision a(0:9)
Double precision b(0:5,0:67)
Integer argos(100)
Integer WEDNESDAY
Parameter(WEDNESDAY=134)
Double precision a(0:39)
	float a[30], b[4];
	double a[10];
	double b[6][68];
	int argos[100];
	const int WEDNESDAY= 0;
	
	const int dim = 40;
	double a[dim];

Dimension b(0:2), A(0:1,0:1) b(0) = 0.1 b(1) = 0.3 A(0,0) = 0.1 A(0,1) = 0.2 A(1,0) = 0.3 A(1,1) = 0.4	b[3] = 0.1, 0.3 A[2][2] ={{0.1, 0.2},{0.3, 0.4}};
Stop	exit(1);

ekvivalentne naredbe/kontrole toka

Fortran 77	C

Do i=1,n
a = a+3
b = b+4
End Do
	for (i=1;i<=n;i++)
	{ a = a+3;
	b+=4;
	}

Do i=1,n a = a+3 End Do	for (i=1;i<=n;i++) a = a+3; /* samo jedna linija */

Do i=j,k,m
. . . . . .
 	
End Do
	for (i=j;i<=k;i=i+m)
	{
	. . . . . .
	}

Do while (i.ne.0)
. . . . .
 	
End Do
	while (i!=0)
	{
	. . . . . .
	}

ekvivalentne naredbe/kontrole toka

Fortran 77	C

If(i.eq.1) then
. . . . . .
End If
	if(i == 1)
	{
	. . . . . .
	}

If(i.eq.1) then
x=3.0
Else If(i.eq.2)
x=4.0
Else
x = 5.0
End If
	if(i == 1)
	x=3.0;
	else if(i == 2)
	x=4.0;
	else
	x=5.0;

If(i.eq.1.and.j.eq.2) then
k = 3
End If
	if(i == 1 && j == 2)
	k = 3;

ekvivalentne matematičke funkcije

Fortran 77	C	Komentar
	#include <math.h>	C header file
y = sqrt(x)	y = sqrt(x)	; dolazi na kraju komande
y = exp(x)	y = exp(x)	
y = log(x)	y = log(x)	
y = log10(x)	y = log10(x)	
y = abs(x)	y = fabs(x)	
y = sin(x)	y = sin(x)	
y = cos(x)	y = cos(x)	
y = tan(x)	y = tan(x)	
y = asin(x)	y = asin(x)	
y = acos(x)	y = acos(x)	
y = atan(x)	y = atan(x)	
y = sinh(x)	y = sinh(x)	
y = cosh(x)	y = cosh(x)	
y = tanh(x)	y = tanh(x)	
z = x**y	z = pow(x,y)	
y = int(x)	y = ceil(x)	najmanji cijeli broj koji nije manji od x

Aritmetičke operacije i Operatori

.	Fortran	Primjer	C/C++	Primjer
zbroj	+	A=B+C	+	a=b+c;
razlika	-	A=B-C	-	a=b-c;
množenje	*	A=B*C	*	a=b*c;
dijeljenje	/	A=B/C	/	a=b/c;
zbroj	+	A=A+B	+=	a+=b
razlika	-	A=A-B	-=	a-=b
+1	+	A=A+1	++	++a,a++

.	Fortran	Primjer	C/C++	Primjer
jednakost	.EQ.	IF(A.EQ.B)...	==	if(a==b)...
nejednakost	.NE.	IF(A.NE.B)...	!=	if(a!=b)...
manje	.LT.	IF(A.LT.B)...	<	if(a<b)...
veće	.GT.	IF(A.GT.B)...	>	if(a>b)...
manje jednako	.LE.	IF(A.LE.B)...	<=	if(a<=b)...
veće jednako	.GE.	IF(A.GE.B)...	>=	if(a>=b)...
logički Not	.NOT.	IF(.NOT.A)...	!	if(!a)...
logički AND	.AND.	IF(A.AND.B)...	&&	if(a&&b)...
logički OR	.OR.	IF(A.OR.B)...	||	if(a||b)...

Operatori

.	Fortran	Example	C/C++	Example
Bitwise AND	IAND	IAND(N,M)	&	n&m
Bitwise OR	IOR	IOR(N,M)	|	n|m
Bitwise Exclusive OR	IEOR	IEOR(N,M)	^	n^m
Bitwise 1's Complement	INOT	INOT(N)	~	~n
Bitwise Left Shift	ISHFT	ISHFT(N,M) (M > 0)	<<	n<<m
Bitwise Right Shift	ISHFT	ISHFT(N,M) (M < 0)	>>	n>>m

Pozivanje fortran funkcija iz C-a

```
#include <stdio.h>
void myfortranfn_ (int *, int *);

main(){

  int ia = 10, ib;
  myfortranfn_ (&ia, &ib);
  printf("\n fortran computed ib=%d \n", ib);
}
```

```
      subroutine myfortranfn(ia,ib)

      integer ia, ib

      ib = 100*ia

      return
      end
```

```
gcc -c mycdriver.c
f77 -c myfortranfn.f
```

```
f77 -o mycdriver mycdriver.o myfortranfn.o
gcc -o mycdriver mycdriver.o myfortranfn.o
```

program nm

- Neki kompajleri ne dodaju znak _ na kraj imena fortran funkcija. Kod f77 kompajlera -fno-underscoring opcija.
- Ovisno o kodu, link proces može biti kompliciran
- nm unix komanda list simbole iz objektnog koda ili datoteke (library).

```
> nm myfortranfn.o
   00000000 T myfortranfn_
```

```
> nm mycdriver.o
        00000000 T main
        U myfortranfn_
        U printf
```

polje kao argument

```
#include <stdio.h>
/* prototype for the fortran test function  */
void myffn_(int *, int *);
main() {
 /* data to be passed to Fortran */
 int in = 2;
 int *idata = (int*) calloc(in, sizeof(int));
 idata[0] = 1; idata[1] = 3;
 printf("C before call: idata[0] = %d \n", idata[0]);
 printf("C before call: idata[1] = %d \n", idata[1]);
 /* call myffn (a F77 function) note underscore */
 myffn_(&in, idata);
 /* printout entries of idata again*/
 printf("C after  call: idata[0] = %d \n", idata[0]);
 printf("C after  call: idata[1] = %d \n", idata[1]); }
```

mydriver1.c i myffn.f

```
c     SOURCE myffn.f
      subroutine myffn(in, idata)
c     indicate types of function arguments
      integer in
      integer idata(*)
c     print out idata entries
      write(*,*) 'f77 before change: idata(1) = ', idata(1)
      write(*,*) 'f77 before change: idata(2) = ', idata(2)
c     access and change idata contents (note indexing from 1)
      idata(1) = idata(1)+1
      idata(2) = idata(2)+3
c     print out idata entries
      write(*,*) 'f77 after  change: idata(1) = ', idata(1)
      write(*,*) 'f77 after  change: idata(2) = ', idata(2)
      return
      end
```

Pozivanje C funkcije iz F77

Fortran file:

```
      Program Callc
      INTEGER CR2
      N=10
      CALL CR1(N,M)
      WRITE(6,20) N,M
   20 FORMAT(' The square of',I3,' is',I4)
      K=CR2(N)
      WRITE(6,30) N,K
   30 FORMAT(' The cube of',I3,' is',I15)
      CALL EXIT
      END
```

C file:

```
void  cr1_(int *n, int *m)
{
   // Compute the square of n, return in m
   int k;
   k=*n;
   *m=k*k;
   return;
}
int  cr2_(int *n)
//  Compute the cube of n
{
   int m,k;
   k=*n;
   m=k*k*k;
   return m;
}
```

f77 -c myfortprog.f

gcc -c mycfn.c

f77 -o myfortprog myfortprog.o mycfn.o

Recept za miješanje

- Recept f77 & C
  - **nm** otkriva kojim simbolom je kompajler označio funkciju u objektnom kodu
  - f77 koristi pointere, argumenti su pointeri na varijable
  - Ako je argument u f77 podprogramu polje, trebamo pointer na prvi član u polju (a ne pointer na pointer)
  - ne koristimo _ (underscore) u f77 jer kompajler može dodati još
  - U C jeziku možemo dodati _ ako funkciju pozivamo iz f77

Makefile

Rastavljen je na dva dijela:

- Make.opts
- Makefile

1)Make.opts odabir kompajlera, linkera i datoteka

```
# specify which compilers to use for c, fortran and linking
CC      = gcc
FC = g77
LD      = gcc
LDF     = g77
# compiler flags to be used (set to compile with debugging on)
CFLAGS          =  -I$(HDRDIR) -g
LIBS    = -L. -L../lib  -llapack -lcblas -lblas -lg2c -lm
LIBSF  = -L. -L../lib  -llapack -lcblas -lblas
```

Makefile

2. Makefile

Naredba make kompajlira i linka sve

make cfortran napravi primjere koji se odnose na C i fortran (do ovog teksta)

make cblas napravi primjere za BLAS iz C jezika

make fblas napravi primjere za BLAS iz F77

```
# rule for .c files
.c.o:
        $(CC) $(CFLAGS) -c $<
all: cfortran cblas fblas
OBJ1 = mycdriver.o myfortranfn.o
fblas: blaslevel1 blaslevel2
mycdriver:  $(OBJ1)
        $(LD) -o mycdriver $(OBJ1) $(LDFLAGS) $(LIBS)
```

Komentar

Implicitno kompajliranje

Svi programi

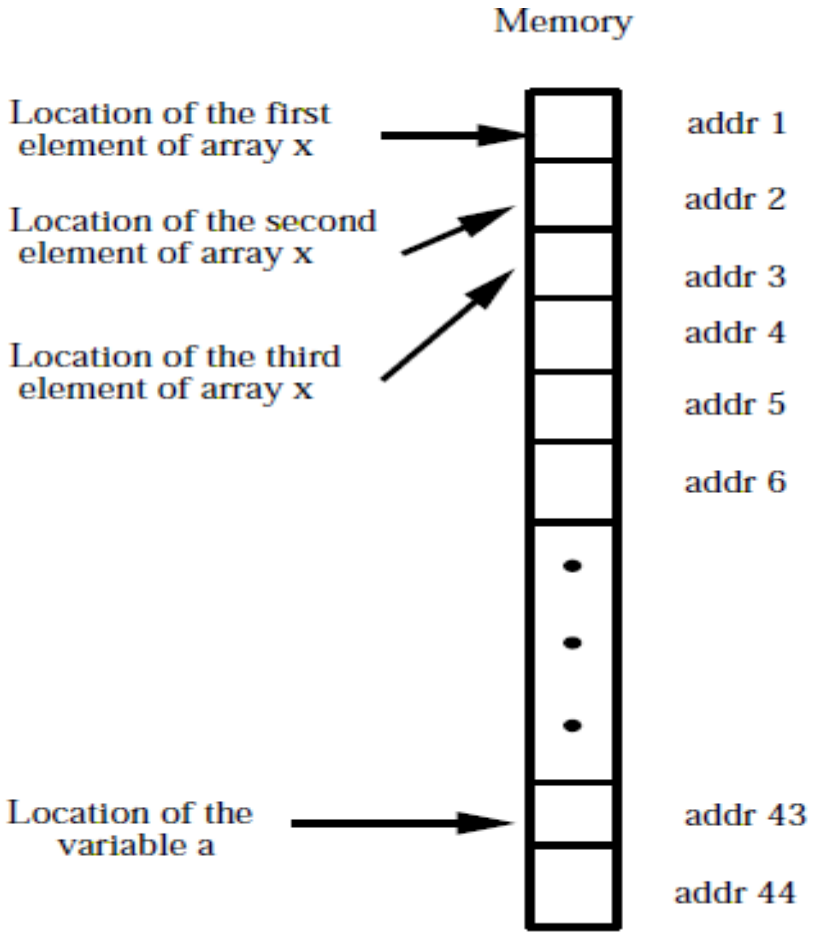
Objektni kod za 1. program

Dio programa

1. program

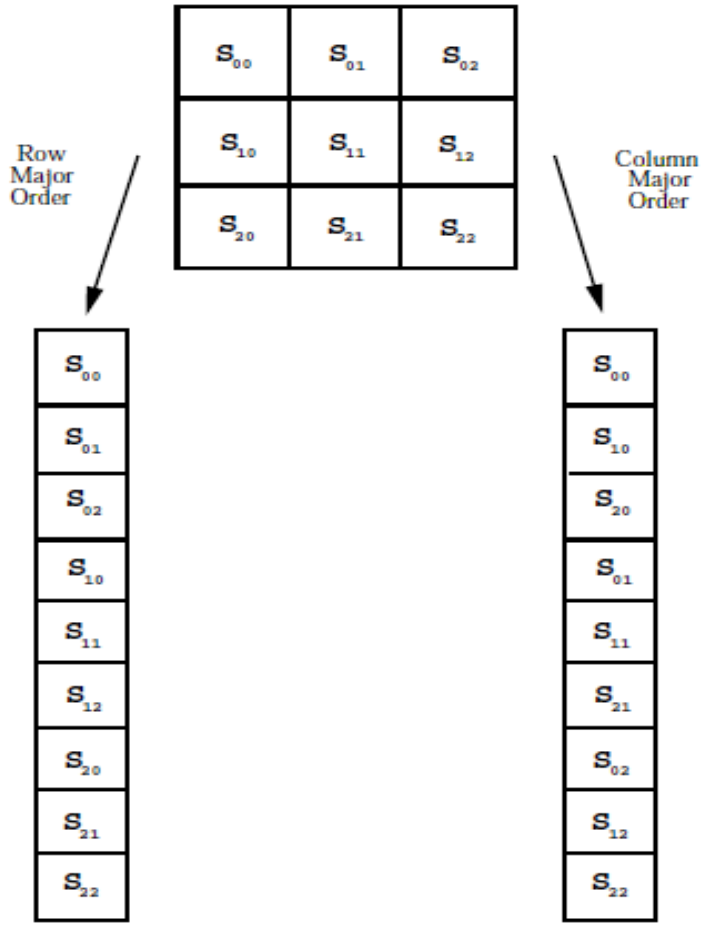
Memorija, matrice

- adresa u memoriji
  - addr2=addr1+addrset
    - addr mjesto gdje se nalazi memorija
    - addrset je offset

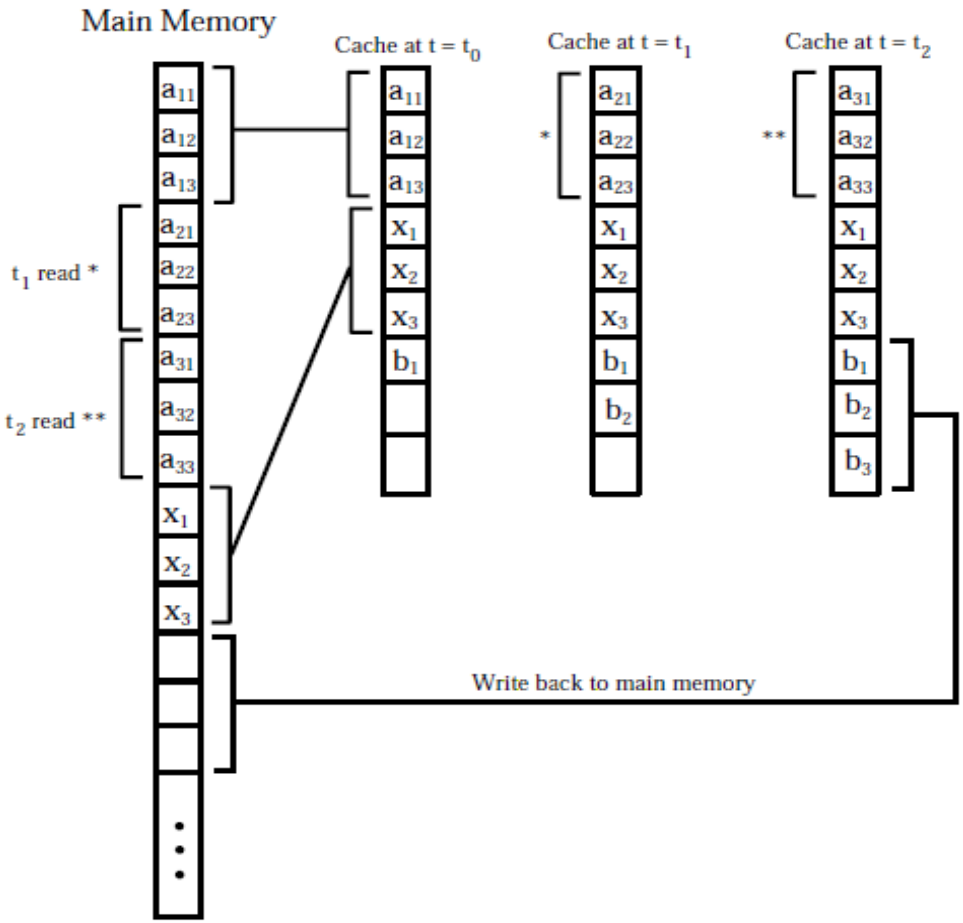


Matrice, memorija

- Row major i Column major order
  - veličina memorije ista
  - linearno uređenje je različito
  - Za simetrične matrice, isto uređenje
  - C koristi row major
  - Fortran koristi column major
- Glavna memorija CPU (spora)
  - Cache: Level 1 (L1) i Level 2 (L2)



- Ideja je da se cache memorija koristi što više kad se jednom napuni (cache hits, cache miss)
- Cache bloking maksimalno iskorištenje cache strukturom podataka i operacija npr. odabirom row major uređenja.
- brzina ovisi o algoritmu koji koristi raspored memorije u cache memoriji



Literatura

- Online literatura:
  - Numerička matematika-osnovni udžbenik, PMF, projekt mzt.
  - C Pozirkidis: C++ for Fortran Programmers, lecture notes
  - George W. Collins, II:
    Fundamental Numerical Methods and Data Analysis
  - Fortran and C/C++ Mixed Programming
  - Calling Fortran Subroutines from C/C++
  - USER NOTES ON FORTRAN PROGRAMMING (UNFP)

Zadaci za praktikum

(a) Napravite programe pomoću make naredbe: make cfortran

(b) Napišite program za množenje 2 vektora.

(c) Napišite program za množenje dviju matrica. Iskoristite programe matrix_mult i readAnddisplay(C/F) (.f,.c). Programi trebaju pročitati matrice iz tekst formata (npr. matrixVector.dat, pročita se dimenzija, a onda matrica), proširite format i dodajte oznaku za komentar npr. *, ili c kao prvi znak.

(d) Napišite program za množenje vektora i matrice.

(e) Pošaljite mail na Aleksandar.Maksimovic@irb.hr s odgovorima u attachmentu.